VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

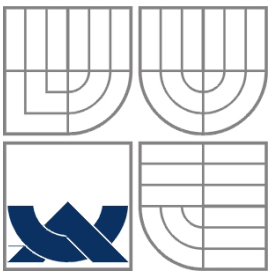
ELEKTRONICKÁ VZDÁLENÁ SPRÁVA DOMÁCNOSTI

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

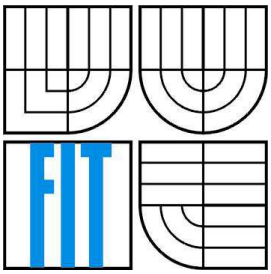
AUTOR PRÁCE
AUTHOR

JAN ALTRICHTER

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ELEKTRONICKÁ VZDÁLENÁ SPRÁVA DOMÁCNOSTI

ELECTRONIC REMOTE HOUSEHOLD CONTROL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAN ALTRICHTER
AUTHOR

VEDOUCÍ PRÁCE — Ing. DAVID KUBÁT
SUPERVISOR

BRNO 2012

Abstrakt

Tato bakalářská práce popisuje vytvoření modelu systému pro vzdálenou správu domácnosti a inteligentní domácnosti. Zaměřuje se zejména na bezpečnost hlídaného objektu a jednoduché vzdálené ovládání připojených zařízení. Jako centrální řídící jednotka je použit FITkit, ke kterému jsou připojeny zařízení, a ke komunikaci používá ethernetový modul. Vytvořený systém je vyvinut pro práci na operačním systému Windows. Součástí práce je fyzicky realizovaný model systému.

Abstract

The thesis describes the creation of an electronic remote household control system and intelligent household. It focuses on security of the guarded object and simple remote control of connected devices. Devices are connected to FITkit which is used as the central control unit; an Ethernet module is used for communication. The created system is developed for Microsoft Windows operating system. One part of the thesis is created by the physical model of the system.

Klíčová slova

Inteligentní, budova, FITkit, zabezpečení, systém, model, vzdálená správa, senzor.

Keywords

Intelligent, building, FITkit, security, system, model, remote control, senzor.

Citace

Altrichter Jan. *Elektronická vzdálená správa domácnosti*, bakalářská práce, Brno, FIT VUT v Brně, 2012

Elektronická vzdálená správa domácnosti

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Davida Kubáta. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jan Altrichter
16. Května 2012

Poděkování

Rád bych poděkoval vedoucímu práce Ing. Davidu Kubátovi za jeho čas věnovaný pomoci při řešení této práce. Dále bych rád poděkoval mému kamarádovi Marku Skopalovi, který mi poskytl konzultace při řešení webové části.

Obsah

Obsah 1
1 Úvod 3
2 Inteligentní budovy 4
3 Popis technologií 6
3.1 Detektor kouře 6
3.2 Pohybové čidlo 7
3.3 Magnetický spínací kontakt 8
3.4 Dvoustavový přepínač 8
3.5 Elektromagnetické relé 8
3.6 IP kamera 9
3.7 Centrální řídící jednotka 10
3.7.1 FITkit 10
3.7.2 Ethernetový modul pro FITkit 10
3.8 Zařízení pro síťovou komunikaci 11
4 Návrh řešení 12
5 Implementace 14
5.1 Technologie použité při vývoji 14
5.1.1 HTTP protokol 14
5.1.2 PHP 14
5.1.3 XHTML a CSS 15
5.1.4 JavaScript 16
5.1.5 NETTE Framework 16
5.1.6 MySQL 16
5.1.7 Jazyk C a VHDL pro FITkit 17
5.2 Bezpečnost 17
5.3 Realizace na FITkitu 18
5.4 Realizace webové aplikace 19
5.4.1 Popis komunikace 19
5.4.2 Časové spouštění, termostat a seznam změn stavů 19
5.5 Popis databázové struktury 20
5.6 Vizuální reprezentace systému 20
5.7 Zprovoznění a zavedení systému 21
5.8 Použité zařízení a technologie 22
5.9 Popis komunikačního protokolu 22

5.10 Popis modelu vytvořeného systému 23
6 Popis zdrojových kódů 24
6.1 Řídící jednotka 24
6.1.1 MCU jádro FITkitu 24
6.1.2 Popis hardwaru v FPGA 24
6.2 Webový server 25
7 Testování a možná rozšíření 28
7.1 Testování vytvořeného systému 28
7.2 Možná rozšíření 28
8 Závěr 30

1 Úvod

Tématem práce je vzdálená správa domácnosti, což je stále více se rozšiřující trend v oblasti zabezpečení a ovládání domácnosti. Domácnosti vybavené touto technologií mohou být řízeny a monitorovány vzdáleně. Popisovaná technologie disponuje možnostmi zjistit v nepřítomnosti, zda například nezůstaly otevřené dveře, nebo umožní vypnout světlo, pokud odejdeme z domu a zapomeneme je zhasnout. Díky možnosti dálkového ovládání je život v domácnosti jednodušší a pohodlnější, což každý obyvatel zajisté ocení. Domácnost vybavenou systémem pro vzdálenou správu, lze také nazvat jako inteligentní domácnost.

Cílem této práce bylo vytvořit funkční model a webovou aplikaci, které dohromady tvoří systém pro vzdálenou správu domácnosti. Vytvořený systém je v jádru srovnatelný s dnes již rozšířenými inteligentními domy, pouze se nezabývá vytvářením různých místních scénářů ale spíše vzdálenou kontrolou objektu. Systém by měl umožnit vlastníkovi objektu odkudkoli, kde je dostupný internet, zjistit stav přístupových cest, zajistit vizuální kontrolu a v menší míře ovládat některá zařízeni.

Systém požadovaného typu by měl splňovat podmínku jednoduchosti a intuitivnosti ovládání i pro počítačově méně gramotné uživatele, kteří nemají zájem hledat různá nastavení v zanořených menu a podobné věci. Naopak pokud má takový uživatel k dispozici jednoduchý plánek objektu, kde vidí vše potřebné na jednom místě, je spokojený a dokáže systém plně využívat.

Druhá kapitola obsahuje teoretický úvod do problematiky inteligentních budov a běžně používaných bezpečnostních systémů. Další kapitola popisuje technologie používané při vzdálené správě domácnosti, které jsem dle zadání nastudoval a vybrané detailně popsal a doplnil vhodnými obrázky. V kapitole s pořadovým číslem čtyři je popsán návrh celého vytvářeného systému. V páté kapitole je popsána implementace celého systému. Tato kapitola je rozdělena do deseti podkapitol. První podkapitola se zabývá popisem použitých vývojových technologií. Zbývající podkapitoly již popisují vytvořenou aplikaci a jsou děleny tak, jak jsou rozděleny části vytvořeného systému, dále popisem vizuální stránky, popisem zavedení systému do provozu, popisem komunikačního protokolu a také popisem fyzicky vytvořeného modelu systému. Šestá kapitola popisuje detailně použité funkce, jejich vazby a umístění v souborech. Kapitola sedmá popisuje testy provedené na modelu systému, jejich průběh a jejich výsledky. Poslední kapitolou je závěr a zhodnocení výsledků.

2 Inteligentní budovy

Systémy pro vzdálenou správu domácností nejsou v České republice velmi rozšířené, jelikož cena nevyváží přínos systému pro zákazníka. Cena takového systému se pohybuje v řádech statisíců, při instalaci do stávající domácnosti (samozřejmě je vše závislé na velikosti a počtu ovládaných zařízení). Pokud instalace probíhá zároveň se stavbou domu, tak se cena pohybuje zpravidla v rozmezí 4-8% z ceny za celou stavbu.

Domy, do kterých je nainstalován systém pro vzdálenou správu, jsou většinou doplňovány programem, který se snaží o umělou inteligenci. Takové domy jsou poté nazývány inteligentními. Inteligentní dům se dokáže chovat ekologicky, učí se zvyklosti uživatele a podle toho dokáže například snížit náklady na vytápění, když ví, že uživatel nebude ještě nějakou dobu doma.

Inteligentní budovy obsahují vždy centrální řídící jednotku, sloužící jako mozek celého objektu. K tomuto zařízení jsou potom připojeny další moduly. Každý z modulů se stará o něco jiného, například modul pro ovládání osvětlení, modul pro regulaci tepla a další. Jednotlivé moduly mají k sobě připojená koncová zařízení. Mezi takové zařízení patří vypínače, zásuvky, elektrické regulační ventily a senzory. Vzdáleně ovládané prvky pracující na principu přepínačů (vypínače, regulační ventily) lze ovládat vzdáleně i lokálně, proto nelze využít dosud používanou technologii. Tyto prvky jsou navrženy speciálně pro účely inteligentních domů a jejich použití se předpokládá s odpovídajícím modulem.

Inteligentní budovy používají velké množství senzorů, které se dělí na citlivé prvky, transducery a transmittery. Citlivé prvky jsou takové, které reagují na změnu okolí změnou elektrického odporu ne napětí. Transducery jsou prvky posílající elektrický signál se změnou na citlivých prvcích. Transmittery jsou prvky reagující na fyzické podměty a mohou být použity jako vstupní zařízení pro připojení do kontrolní jednotky. V praxi jsou transducery a transmittery často kombinovány a tím vznikají stavové a analogové senzory (aktory). Stavové senzory mají pouze dva stavy (stav zapnuto a stav vypnuto). Analogové senzory převádí hodnoty na elektrický signál a dělí se na pasivní a aktivní. Stav pasivních senzorů je zjišťován z centrální jednotky a nepotřebují žádný další zdroj napětí. Oproti tomu aktivní vyžadují přídavný zdroj napětí a signál vychází ze zařízení do řídící jednotky.

V případě nasazení technologie inteligentních budov do domácností přibyly požadavky na propojení systémů používaných v komerčních budovách se systémy využívanými hlavně v domácnostech. Mezi takové systémy patří rozvod zvuku po celém domě s možností centrálního i vzdáleného ovládání, dále připojení všech multimediálních zařízení k jednomu centrálnímu, možnost ovládat celý dům pomocí jednoho bezdrátového zařízení, například pomocí chytrého telefonu a další.

Pro zabezpečení objektů se běžně používají centrální zabezpečovací systémy. Tyto systémy kombinují kontrolu přístupových cest s pohybovými čidly. Tyto systémy bývají vybaveny GSM bránou, pro odeslání SMS zprávy o narušení objektu, nebo bývají připojeny na centrálu příslušné bezpečnostní služby, která zařídí vše potřebné při narušení objektu. Tyto stávající zabezpečovací systémy je možné přirovnat k předchůdcům inteligentních domů. Při implementaci inteligentních funkcí do staršího domu tak lze využít stávající bezpečnostní systém a obohatit ho o další funkce. S přispěním internetu je možné vcelku jednoduše sledovat stav objektu z jakéhokoli místa na světě.

K vypracování této kapitoly byly použity zdroje [1] a [2]. Dále jsem za účelem získání co nejvíce informací o tématu inteligentních domů, navštívil mezinárodní veletrh elektrotechniky, elektroniky, automatizace a komunikace Amper. Veletrh Amper se konal ve dnech 20-23.3.2012 na Brněnském veletržním výstavišti. Zde jsem navštívil stánky firem Elkop, Foxtrot a ABB, které patří mezi špičkové firmy zabývající se inteligentními domy v Česku. Se zástupci firem jsem

prokonzultoval veškeré informace týkající se jejich produktů, jak po technické stránce, tak i po stránce instalace do budov. Na základě získaných informací jsem upravil část své stávající práce do podoby běžně užívané u těchto firem.

3 Popis technologií

Při tvorbě jakékoli práce je před začátkem návrhu potřeba dohodnout použité technologie. Před výběrem technologií bylo nutné analyzovat zadání a vybrat nejvhodnější a nejdostupnější technologie, které by splňovaly všechny požadavky a byly cenově přijatelné. Analýza probíhala po konzultaci se zadavatelem práce, kterého v tomto případě představuje škola, zastoupená vedoucím této práce.

Z analýzy vyplynulo, že bude potřeba použít hardwarové zařízení s vlastním operačním systémem, které bude realizovat komunikaci mezi čidly a počítačem. Po detailní analýze vhodných zařízení vyhověl nejlépe FITkit, který je dostupný pro studenty zadarmo a je k němu dostatek periferií.

Nyní zbývá už jen propojení obou částí dohromady. Jelikož FITkit disponuje možností připojit ethernetový modul a serverová a klientská část komunikují síťově, bylo nejvhodnější využít HTTP protokol, který je standardizovaný, místo vymýšlení vlastního protokolu.

3.1 Detektor kouře

Detektory kouře jsou různých typů, a to detektor ionizační, optický, hlásič teplot a plamene. [15]

Detektor ionizační detekuje změnu vodivosti vzduchu, ke které dojde úbytkem ionizovaných částí vzduchu v měřící komoře. Tento typ hlásiče detekuje viditelný i neviditelný kouř a je velmi citlivý a levný. Funguje na principu měření vodivosti vzduchu v komoře, kde je vzduch ionizován radioaktivním prvkem. Velkou nevýhodou tohoto principu měření je přítomnost radioaktivního prvku a jeho likvidace.

Dalším typem je detektor optický. Tento detekuje změnu odrazivosti částic vzduchu při hoření. Toto zařízení obsahuje infračervenou diodu a fototranzistor, které jsou umístěny tak, aby nebyly z venku přímo viditelné. Detektor pracuje tak, že pokud vnikne kouř dovnitř do měřící komory, tak dojde k odrazu infračerveného světla zpět na fototranzistor a tím dojde k sepnutí připojeného obvodu, pomocí kterého je možno například spustit alarm nebo signalizační zařízení. Různí výrobci využívají různé způsoby zpracování měření a různé tvary komor, podle druhu očekávaného kouře. Nejčastěji jsou používány dva systém s různými tvary, na rozeznání jak temných, tak světlých kouřových částic.

Často používanými jsou také hlásiče termostatické a termodiferenciální. Termostatický hlásič detekuje překročení teploty na sledovaném místě. Teplota spuštění poplachu je nastavena výrobcem. Většinou se používá tam, kde je normální výskyt částic kouře ve vzduchu, které při tom nemusí nutně znamenat požár, například v kuchyni, kde také při vzniku požáru dojde k velkému nárůstu teploty. Druhým typem teplotního hlásiče je hlásič termodiferenciální, který detekuje rychlost nárůstu teploty ve sledovaném místě. Pokud nárůst teploty překročí za daný čas nastavenou hranici, dojde k sepnutí poplachového obvodu a tím k vyhlášení poplachu. Jeho použití je prostředím podobné termostatickému, pouze je vhodnější tam, kde je nárůst teploty mnohem rychlejší.

Posledním zmíněným je hlásič plamene. Toto čidlo detekuje výskyt světelných projevů typických pro hoření, v infračervené nebo ultrafialové oblasti. Vnitřní uspořádání takového čidla je podobné jako v kameře, která sleduje hlídané místo. Tento senzor reaguje na změnu intenzity sledovaného záření v čase a ne na jeho intenzitu. Používané senzory citlivé na záření jsou používány různé podle používaného prostředí. Pro vnitřní objekty se používají čidla se senzorem citlivým na ultrafialové záření a pro venkovní a těžké průmyslové objekty je používáno infračervené záření.

Mnou použitý detektor kouře Longhorn LH-87, je dle specifikace [3], napájen 12 V stejnosměrného napětí nebo 220 V střídavého napětí s frekvencí 50Hz. Detektor se připojuje pomocí pěti vodičů různých barev, a to:

- černý vodič pro připojení země
- červený vodič pro připojení plusového napětí 12 V
- zelený vodič pro připojení alarmu, který je sepnut v krizovém stavu
- modrý vodič pro připojení alarmu, který je sepnut v klidovém stavu
- žlutý vodič, který slouží jako párový pro oba alarmové vodiče (zelený, modrý).

Obrázek 1: Pohybové čidlo. Příklad běžně používaného detektoru (vlevo). Schéma zapojení použitého modelu (vpravo).

3.2 Pohybové čidlo

Nejvíce rozšířené druhy pohybových čidel jsou pyroelektrické a detektory reagující na odražené záření, které samy vysílají. [16]

Pyroelektrické jevy generují elektrický náboj, závislý na tepelném toku procházejícím tělesem senzoru. Pohybující se objekt vyvolá rozdíl teplot, který způsobí mechanickou deformaci. Tato deformace je způsobena absorpcí záření z objektu. Když čidlo zareaguje na tepelné změny v detekčním poli, dojde k sepnutí připojeného obvodu, takzvaně dojde k vyvolání poplachu.

Dalším druhem jsou čidla pracující na principu sonaru, vysílají ultrazvukové vlny a přijímají jejich odraz. Sonarovým čidlem je také možno měřit vzdálenost, polohu a zrychlení sledovaného objektu od čidla. Tato čidla se v jednodušších zabezpečovacích systémech nevyužívají z důvodu nutnosti přídavného zařízení, které vyhodnotí výsledky měření.

Mnou použitý detektor pohybu Longhorn, typu PDS-901A, dle specifikace [4] může být napájen 9-16 V stejnosměrného napětí. Pohybové čidlo se připojuje pomocí svorkovnice přiletované na základní desce uvnitř čidla. Svorkovnice obsahuje tři dvojice kontaktů. První dvojice je pro připojení vstupního napětí. Další je spínač pro alarm při pohybu ve sledované oblasti a poslední je spínač pro alarm spouštěný při násilném vniknutí do obalu čidla.

Obrázek 2: Pohybové čidlo. Příklad běžně používaného čidla (vlevo). Schéma zapojení použitého modelu (vpravo).

3.3 Magnetický spínací kontakt

Magnetický spínací kontakt pracuje podobně jako klasický spínač, kdy je na něj připojený obvod v jednom stavu rozpojený a ve druhém stavu sepnutý. Magnetický kontakt je na rozdíl od klasického spínače spínán přiblížením magnetu, což je druhá část kontaktu, k samotnému spínači. [19]

V zabezpečovacích systémech se tento kontakt používá k zajištění přístupových cest, a to tak, že samotný spínač je připevněn na rámu dveří, respektive okna a magnet, který spínač aktivuje, je umístěn na dveřích, respektive okně. Celý komplet bývá dodáván zatavený v plastu s připravenými otvory pro přišroubování.

Mnou vybraný magnetický kontakt značky Longhorn má z části umístěné na rámu okna nebo dveří vyvedené dva samostatné vodiče, které mohou být libovolně připojeny do systému, který mají spínat. Specifikace spínače dovoluje spínat maximálně 100 V s protékajícím proudem až 500 mA. Tento spínač má spínací vzdálenost 15 až 25 mm. Specifikace kontaktu byla převzata zde [5].

3.4 Dvoustavový přepínač

Dvoustavový přepínač je speciálním druhem spínače. Přepínač se chová jako dva spínače s jedním společným vývodem. Tyto dva spínače pracují do kříže, a to tak, že když je první ve stavu sepnuto, tak druhý je ve stavu rozepnuto a naopak. V praxi to znamená, že když připojíme na jeden vstup zem a na druhý vstup referenční napětí, dosáhneme přepínáním toho, že se na výstupu střídá zem s referenčním napětím a tím dosáhneme například rozsvícení a zhasnutí žárovky. Pokud u tohoto přepínače využijeme pouze jeden vstup, dá se použít jako obyčejný spínač, čehož je v praxi často využíváno. Pokud daný přepínač otočíme, můžeme ho použít k přepínání mezi dvěma různými obvody. [19]

Dvoustavový spínač značky Zippy [6] osazený páčkou, dovoluje spínat 125 V a protékající proud maximálně 3 A. Spínač má tři konektory, z nichž první je podle polohy páčky propojen buď s druhým nebo s třetím. Zařízení vykazuje přechodový odpor 30-100 mΩ, což je pro účely použití v popisovaném systému zanedbatelná hodnota.

3.5 Elektromagnetické relé

Relé je elektrická součástka. Tato součástka obsahuje přepínač, respektive přepínače, které jsou ovládány elektromagnetem. Elektromagnetické relé je příkladem využití elektromagnetu v automatizovaných součástkách a řídících systémech. Relé obsahuje elektromagnet a pohyblivou kotvu. Elektromagnet je tvořen cívkou a jádrem s magneticky měkké oceli. Kotva je taktéž vyrobena z magneticky měkké oceli a je uložena volně, tak aby mohla být přitahována elektromagnetem. Hlavním principem činnosti je přitahování kotvy elektromagnetem v době, kdy jím prochází elektrický proud, který jej zmagnetizuje. Takto přitažená kotva rozepne jeden kontakt a sepne kontakt druhý, takovému relé se říká přepínací. K přitažení kotvy je potřeba mnohem menší ovládací proud, než je proud, který prochází ovládaným obvodem. Relé lze rozdělit na více typů, a to podle rohů kontaktů na spínací, rozpínací a přepínací. Dále je lze dělit například podle elektrických veličin ve spínaném i spínacím obvodu a podle principu činnosti (bistabilní nebo se zpožděným přítahem). [19]

Obrázek 3: Magnetický spínací kontakt (vlevo). Mikrospínač (uprostřed). Elektromagnetické relé (vpravo).

3.6 IP kamera

Je to zařízení, které se skládá z webkamery, síťového prvku a volitelně může obsahovat enkodér videa. Síťový prvek se stará o fyzické připojení do sítě a většinou pokrývá alespoň tři vrstvy ISO/OSI modelu (fyzickou, spojovou a síťovou vrstvu). [17]

Pokud kamera provádí kompresi videa, musí obsahovat enkodér. Komprese videa je nutná například kvůli zálohování nebo menším nárokům na síť, na které pracuje. Pokud kamera neobsahuje enkodér, musí být kódování videa prováděno na zálohovacím počítači, a to buď hardwarově nebo softwarově.

Na zařízení většinou jedou různé podpůrné servery, například webový server, FTP server, nebo server pro proudové vysílání. IP kamera se připojuje do ethernetové sítě a každá má svoji vlastní IP adresu. Kamera komunikuje s ostatními zařízení v síti například pomocí HTTP protokolu a dá se k ní přistoupit například pomocí webového prohlížeče. V případě přístupu pomocí webového prohlížeče musí kamera podporovat webový přístup.

Většina IP kamer je vybavena PoE (Power over Ethernet) napájením. Takovéto zařízení je realizování po ethetnetovém kabelu, který v tomto případě vystupuje jako datový. Napájení je realizováno pomocí speciálního zdroje, do kterého je přiveden datový vstupní ethernetový kabel. Jako výstup slouží také ethernetový kabel, který je kromě dat obohacen o napájení.

IP kamery se využívají na monitorování objektů a pro živé náhledy na internetových stránkách. Jejich výhodou je možnost napájení přes datový kabel, díky čemuž odpadá nutnost vést další kabel s napájením a také je jednodušší záloha systému pomocí centrální UPS.

Použitá IP kamera [7] je napájena přes ethernet zdrojem generujícím 24 V s odběrem proudu 500 mA. Zdroj je napájen 230 V střídavého napětí s frekvencí 50Hz. Kamera pracuje při teplotě -40°C až +70°C, při relativní vlhkosti 20-80%. Tyto podmínky jsou absolutně vyhovující pro zamýšlené místo využití. Tato kamera dovoluje použít rozlišení až 1280 x 800 px, s kompresí pomocí kodeku H.264. Pro přenos dat disponuje kamera Fast Ethernetem 100Base-T.

Obrázek 4: IP kamera.

3.7 Centrální řídící jednotka

Centrální řídící jednotka je srdcem celého systému. Tato jednotka provádí digitalizaci analogových spínačů a kontaktů a odesílá jejich aktuální stav přes ethernet dále. Řídící jednotka CPU je realizována pomocí FITkitu.

3.7.1 FITkit

FITkit [8] je samostatný hardware s mikrokontrolérem, hradlem FPGA a řadou periferií, jako je klávesnice, či display. FITkit lze neomezeně modifikovat, obdobně jako software na počítači. Při návrhu aplikací se využívá možnosti popsat hardware pomocí programovacího jazyka VHDL, díky čemuž je popis velmi podobný návrhu softwaru. Generování programovacích dat pro FPGA z VHDL probíhá pomocí profesionálních návrhových systémů. Software pro mikrokontrolér se tvoří v jazyce C a překládá se pomocí GNU překladače.

V mém případě je FITkit využit pro obsluhu spínačů, senzorů a vypínačů. Spínače jsou připojeny na vstupně-výstupní rozhraní JP10, na portech 23-52. Zařízení jsou připojena k jednotlivým portům a po sepnutí přivedou na daný port logickou 0. Logická 0 v mém případě reprezentuje stav sepnuto, protože pokud je port odpojen, je reprezentován logickou 1. Tuhle drobnost je možné odstranit dvoustavovým přepínačem, ale u čidel a magnetických spínačů by bylo potřeba využít relé, což je náročnější na realizaci a cenu.

JP10, které komunikuje s čidly, je připojeno pouze k FPGA. Naopak pro komunikaci po přes http protokol je potřeba realizovat komunikaci z MCU. Aby bylo možno komunikovat s čidly z mikrokontroléru, je potřeba použít sběrnici SPI.

3.7.2 Ethernetový modul pro FITkit

Je to modul, který rozšiřuje FITkit verze 2.0 o možnost komunikovat pomocí rozhraní Ethernet. K modulu je k dispozici knihovna *linenc28j60*, obsahující implementaci protokolů IP, ARP, UDP,TCP a ICMP. K mé práci využívám protokol TCP, nad kterým pracuje HTTP protokol, využívaný pro komunikaci mezi FITkitem a webovým severem.

Ethernetový modul obsahuje, řadič 10 Mb Ethernetu, komunikující pomocí SPI. Dále obsahuje konektor RJ45 pro připojení do LAN obsahující oddělující transformátor a dvě diody indikující stav zařízení. Modul je připojen k FITkitu pomocí dvaceti pinového konektoru na vstupně-výstupním rozhraní JP10 na portech 1-20.

Obrázek 5: FITkit (vlevo). Ethernetový modul pro FITkit (vpravo).

3.8 Zařízení pro síťovou komunikaci

Důležitou součástí systému je komunikace po ethernetové síti. K této komunikaci musí být použito multifunkční síťové zařízení, které zvládá v minimálním případě funkci přepínače a DHCP serveru. Při nasazení do reálného provozu je vhodné, aby zařízení disponovalo dále možností přístupu do sítě WAN, k čemuž je nutné použít router, či modem dle poskytovaného připojení. Další výhodou takového zařízení pro použití v popisovaném systému, je přístupový bod pro bezdrátovou síť, usnadňující rozvod kabeláže. V případě výroby modelu systému jsem použil multifunkční zařízení WIRELESS N 150 HOME ROUTER od firmy DLINK. Toto zařízení obsahuje všechny výše požadované aspekty a je cenově dostupné.

4 Návrh řešení

Po nastudování tématu problematiky inteligentních bodnuv a výběru vhodných technologií bylo nutné vytvořit návrh výsledného systému. Zadání vyžaduje vytvoření webové aplikace se zabezpečeným rozhraním. Pro zabezpečení jsem se rozhodl použít autentizaci pomocí uživatelského účtu a šifrovaný protokol https pro komunikaci.

V dalším kroku bylo nutné zvolit řídící jednotku, která zvládne ovládat zvolená zařízení a komunikovat s webovou aplikací. Po prostudování dostupných zařízení jsem zjistil, že bude nejlepší využít FITkit s možností připojení ethernetového modulu, který dovoluje komunikovat stejně jako webová část systému a to pomocí http protokolu. Dále jsem zjistil, že pro komunikaci bude nutné mírně upravit http protokol.

Po prostudování dokumentace FITkitu je potřeba zvolit nejvhodnější sběrnici vyhovující všem vybraným zařízením. V rámci výběru sběrnice je nutné přemýšlet také nad předáváním získaných stavů přes ethernetový modul webové aplikace. Zde se otevírají dvě cesty řešení tohoto problému. První cestou je komunikace se sběrnicí přímo s MCU, které posílá data přes síť do webové aplikace přes ethernetový modul. Druhou cestou je použití FPGA, které komunikuje přes sběrnici transparentnějším způsobem, ale je nutné předat data do MCU. Komunikace mezi MCU a FPGA lze dosáhnout pomoc sběrnice SPI pomocí jednoduchého adresování. Vyzkoušel jsem obě možnosti a nakonec kvůli lepší funkčnosti zvolil variantu druhou s připojením sběrnice označené JP10.

V tuto chvíli je vyřešena komunikace, bezpečnost i centrální řídící jednotka a zbývá navrhnout samotné zabezpečení objektu a ovládání zařízení. To si rozdělíme na více částí, a to zabezpečení přístupových cest, vizuální kontrolu, kontrolu pohybu nežádoucích osob, kontrolu vzniku požáru a úniku plynu a vzdálené ovládání. Každá z těchto částí používá jiný druh detekce, proto je nutné uzpůsobit celý systém tak, aby zvládl obsluhovat všechny tyto typy. Díky vhodnému výběru zařízení je možné až na vizuální kontrolu získávat stav stejným způsobem, a to reprezentovaný dvěma logickými hodnotami *true* a *false*. Tyto hodnoty poté znamenají pro každé zařízení jiný stav, ale to je možné jednouchým algoritmem sjednotit. Pro vzdálené ovládání je situace obdobná a díky vhodně zvolenému mezičlánku v podobě relé, jsou všechna zařízení ovládána stejným způsobem. Vizuální kontrola je díky zvolení samostatného hardwaru řešena mimo centrální řídící jednotku.

Nyní je nutné vhodně navrhnout funkčnost webové aplikace. Tento návrh musí vycházet z požadavků zadání rozšířených o vhodně zvolené prvky z podobných systému. Aplikace musí umět vhodně zobrazit stavy senzorů, jednoduše ovládat zařízení, uchovávat změny stavů a umožnit parametrizované nastavení spouštění libovolných zařízení. Důležitou vlastností aplikace je komunikace s řídící jednotkou, která musí být vyřešena ajaxově, což umožňuje jazyk JavaScript. Použitím ajaxu se vyhneme častému refreshování stránky a veškerou komunikaci budeme provádět na pozadí.

Uchování změn stavů musí být dobře navrženo, protože při nasazení v běžně velké, každodenně fungující domácnosti, bude dle předpokladů vygenerováno několik stovek záznamů denně. Kvůli této podmínce je nutné využít databázi, která se jeví jako nejvhodnější z možných variant. Při použití databáze, je potřeba požadovaná data jednoduše získat z databáze. Získání dat z databáze je dosaženo pomocí databázových dotazů, které musí být vhodně optimalizovány, aby nedocházelo ke zpoždění při velkém množství dat, což může odpovídat již několikaměsíčnímu používání.

Dalším krokem je navrhnout vhodný způsob zobrazení aktuálních stavů senzorů a ovládání zařízení. Zvolil jsem způsob jednoduchého plánku objektu (v případě této práce odpovídá vytvářenému fyzickému modelu). Plánek obsahuje bloky, jejichž vizuální podoba se změní se

změnou stavu senzorů, nebo po kliknutí na jim vymezenou oblast. Pokud dojde ke kliknutí na vymezenou oblast, nezmění se jen vizuální reprezentace odpovídajícího bloku, ale musí se změnit i stav fyzického zařízení. Této funkčnosti bude dosaženo použitím jazyka XHTML ve spojení s jazykem CSS.

Nakonec v rámci webové aplikace bude nutné zajistit parametrizované spouštění. Při tomto spouštění musí být uchovány nastavené parametry (například čas nebo teplota). K uchování je nejvhodnější použít již dříve zmiňovanou databázi se splněním integritních omezení. K řešení nastavení parametrů a jejich užití je nejvhodnější použít jazyk PHP v kombinaci s jazykem JavaScript.

Poslední částí návrhu je vytvoření fyzického modelu. Rozhodl jsem se vyrobit dřevěný model odpovídající dříve vytvořenému plánu objektu. Model bude obsahovat jednoduchou demonstraci použití magnetických kontaktů při zabezpečení přístupových cest, senzor pohybu, hlásič požáru a úniku plynu, IP kameru pro vizuální kontrolu a LED diody pro demonstraci vzdáleného ovládání. Všechna použitá zařízení připojím pomocí kabeláže a nepájivého pole ke sběrnici řídící jednotky (FITkitu). Centrální řídící jednotku, webovou kameru a webový server propojím pomocí multifunkčního síťového zařízení do jedné ethernetové sítě.

5 Implementace

Po analýze požadavků a výběru vhodných technologií, jsem vytvořil návrh systému a protokolů pro komunikace mezi jednotlivými částmi. Každá část má přesně dané svoje vstupy a výstupy. Webové části jsem navrhnul uživatelské rozhraní, které je jednoduché a výstižné.

Hardwarová část je realizována pomocí FITkitu a čidel, kontaktů, spínačů a přepínačů připojených k němu. Další hardwarovou částí je samostatná IP kamera.
Podstatnou částí systému je serverová část, která se skládá z webového a databázového serveru. Tyto servery mohou být realizovány jako samostatný hardware nebo mohou být pouze virtualizovány na domácím počítači, který se dá využít i k jiným účelů a odpadá tak investice do jednoúčelového zařízení.

5.1 Technologie použité při vývoji

Jelikož má program pracovat vzdáleně, bude nutné pracovat s webovými technologiemi, a to konkrétně PHP na serverové části. Na klientské části se jedná o JavaScript, CSS a HTML. Dále bylo potřeba vybrat nástroj pro tvorbu webových aplikací.

5.1.1 HTTP protokol

Je to standardní protokol internetu, určený pro výměnu hypertextových dokumentů ve formátu HTML. Tento protokol využívá TCP spojení na portu 80. [14]

Tento protokol je typu klient-server a funguje na principu dotaz-odpověď. Jedna strana (klient) pošle druhé straně (severu) dotaz ve formě čistého textu, obsahující název požadovaného dokumentu a dále informace o klientovi. Server tento dotaz zpracuje a odešle odpověď. Tato odpověď obsahuje na prvním řádku výsledek dotazu, následuje hlavička odpovědi ukončená prázdným řádkem. Za hlavičkou již následuje požadovaný dokument.

V mém případě je HTTP protokol využit jak pro interní komunikaci webové aplikace, tak i pro komunikaci mezi webovou aplikací a FITkitem. Pro interní komunikaci zůstává protokol nezměněn, ale pro komunikaci jsem si protokol mírně upravil.

5.1.2 PHP

Hypertextový preprocesor (Původní název Personal Home Page) [9] je skriptovací programovací jazyk. Jazyk PHP je především určen pro tvorbu dynamických stránek, kdy jsou skripty prováděny na serverové straně a k uživateli je přenášen pouze výsledek. Zdrojový kód jazyka PHP se začleňuje do jazyka HTML. Interpret PHP skriptu je možné volat jak pomocí webových služeb, tak pomocí příkazové řádky. PHP je multiplatformní, což znamená, že je nezávislé na použitém operačním systému a skripty je možné přenášet bez jakýchkoli úprav. PHP je syntakticky velmi podobný jako jazyk C a C++.

PHP podporuje mnoho knihoven, například knihovny pro zpracování textu, grafiky a pro práci s databází. Další výhodou je rozsáhlá podpora od poskytovatelů hostingu, což dovoluje výběr této technologie pro vývoj aplikace nezávisle na jejím dalším umístění. Z hlediska bezpečnosti a ochrany autorských práv vývojáře má PHP oproti technologiím jako je JavaScript, který pracuje na straně klienta, výhodu v tom, že pracuje na straně serveru a skripty nejsou uživateli zobrazeny.

Nevýhodou PHP je, že chybí kvalitní ladící nástroje, čemuž se lze vyhnout použitím frameworku.

V mém případě je PHP použito pro práci s databází, pro řešení autentizace a autorizace uživatelů a pro všechny věci, které jsou řešeny dynamicky.

5.1.3 XHTML a CSS

XHTML (extensible hypertext markup language – rozšířený značkovací jazyk) [10] je značkovací jazyk pro tvorbu hypertextových dokumentů, vyvinutý W3C (world wide web consortium). XHTML byl zamýšlen jako nástupce HTML, jehož vývoj se od roku 1999 do roku 2007 zastavil. Jelikož došlo v posledních letech k posunu k nové verzi HTML, nedá se XHTML považovat úplně za nástupce a spíše za paralelní technologii využívanou ke stejnému účelu.

Zatím je vydána pouze verze 1.0, která má tři další verze značené slovně podle způsobu použití. První verzí je verze Strict, která se používá, pokud chceme strukturovaný dokument osvobozený od formátovacích značek souvisejících s rozložením stránky. Další je verze Transitional, umožňující používat překonané tagy a je vhodná pro formátování stránek, které budou zobrazeny starými prohlížeči. Poslední verzí je frameset. Ta přidává stejně jako transitional podporu zastaralých značek a podporu pro rámce.

Dalším důvodem pro rozšíření XHTML byla modularizace. Modularizace přišla s verzí 1.1, kterou schválilo W3C v roce 2001. Tato verze je velice podobná verzi Strict, ke které přidává výhody modularizace.

Zásadním rozdílem mezi XHTML od HTML je jistá benevolence v dodržování syntaxe tagů. Dále je nutné v XHTML dodržovat určenou velikost tagů (psát tagy malými písmeny), jelikož spojením s XML došlo k přidání vlastnosti case-sensitive (citlivost na velikost písmen). Všechny dokumenty musí začínat XML deklarací, čemuž se lze vyhnout kódováním UTF-8 nebo určení kódování vyšším protokolem.

CSS (Cascading Style Sheets – kaskádové styly) [10] je jazyk pro popis způsobu zobrazení webových stránek napsaných v jazycích HTML, XHTML nebo XML. Tento jazyk navrhlo World Wide Web Consortium v roce 1996. Prozatím byly vydány dvě specifikace CSS1 a CSS2. V roce 2011 byla vydána revize druhé verze. Základní myšlenkou kaskádových stylů je oddělit vzhled dokumentu od jeho struktury a obsahu. Zmíněná myšlenka dovoluje oddělený vývoj grafické a funkční části webové stránky nebo změnu jakékoli části bez zásadní závislosti na druhou část.

Dnes nejpoužívanější verzí, je verze 2.1. CSS soubor je tvořen tabulkou kaskádových stylů s pravidly. Každé takové pravidlo obsahuje oddělovač a blok deklarací, které se skládají z vlastností a hodnot přiřazených každé vlastnosti.

CSS oproti formátování pomocí HTML tagů, umí umisťovat prvky na pozici, danou souřadnicemi, relativně k jejich prvku. Toto zajistí vždy umístění prvku na požadované místo. Pro urychlení načtení stránky, si webové prohlížeče ukládají do vyrovnávací cache paměti údaje o rozmístění prvků na stránce, čímž urychlí načítání stránky a také sníží datový tok.

Mezi výhody použití CSS patří jednodušší údržba webové prezentace, kdy například pro změnu barvy nadpisu nemusíme procházet velké množství HTML kódu, ale změní pouze vlastnost v CSS souboru, který je k dané stránce připojen. Další výhodou je již zmiňované oddělení struktury a stylu. Nespornou výhodou je možnost dynamické změny elementů pomocí JavaScriptu, na čemž stojí grafické prezentace webové části popisované aplikace. Pomocí kaskádových stylů je možné vytvořit více stylových předpisů, které se podmíněně použijí podle druhu zobrazovacího zařízení, jako je například počítač, mobilní telefon nebo tiskárna.

5.1.4 JavaScript

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk, zpravidla využívaný jako interpretovaný programovací jazyk pro webové stránky. Většinou se využívá pro interaktivní prvky, nebo pro tvorbu animací. Jádro jazyka je velmi podobné jazykům C a C++, ale podobnost končí na úrovni syntaxe, kde JavaScript má potlačenou typovou kontrolu. [13]

Tento jazyk je často vkládán přímo do HTML kódu stránky. Části JavaScriptového kódu je možné vkládat do HTML pomocí párové značky *<script>* nebo vložením odkazu na soubor, ve kterém je požadovaná část obsažena. Názvem a cestou k souboru se zadává jako parametr značky *<script>*.

Je to klientský jazyk, k jehož fungování není potřeba žádného serveru a může být vykonán i v „offline režimu“ na počítači uživatele. O zpracování skriptů se většinou stará internetový prohlížeč. Díky použití Javascriptu je možné vytvářet takzvané dynamické stránky. Pojem dynamické stránky se používá pro stránky, kde není nutné kvůli sebemenší změně načítat znovu celou stránku, ale pouze potřebnou část. Toto chování umožňuje uživateli pohodlnější a u rozsáhlejších stránek rychlejší práci. V případě, kdy se při načítání stránek stahuje ze serveru větší množství dat, pomáhá JavaScript ke snížení datového toku, což ocení například uživatelé s omezeným datovým limitem.

JavaScript má jako klientský jazyk oproti serverovým velkou nevýhodu v dostupnosti zdrojového kódu. Každý si může zobrazit vytvořený skript a jeho okopírováním porušit autorská práva náležící vývojáři. Další nevýhodou plynoucí z předchozího bodu je to, že JavaScript nedovoluje práci se soubory, aby nedošlo k ohrožení bezpečnosti uživatele.

Popisovaná aplikace nepředpokládá využití zařízení, které nepodporuje JavaScript.

5.1.5 NETTE Framework

Je to softwarová struktura, která slouží jako podpora při programování a vývoji jiných softwarových projektů. Framework může obsahovat podpůrné knihovny, programy nebo návrhové vzory.

Cílem Frameworku je usnadnění práce vývojáři, kdy se framework stará o obecně použitelné věci a vývojář se může plně soustředit na jádro své práce, například při vývoji bankovní aplikace se vývojář stará o správné provedení bankovních transakcí a framework za něj řeší přechod mezi stránkami.

Do základní architektury frameworku spadají frozen spots a hot spots. Frozen spots definují celkovou architekturu, základní komponenty a vztahy mezi nimi. Všechny tyto části zůstávají neměnné. Na rozdíl od toho hot spots vytváří s kódem vývojáře určitou funkcionalitu, z toho důvodu se vytváří dynamicky. V objektově orientovaném prostředí jsou hot spots reprezentovány jako třídy. V případě téhle práce byl zvolen Nette Framework vytvořený Davidem Grudlem [11] a nyní rozvíjený organizací Nette Foundation. Nette je opensorce pro tvorbu webových aplikací v PHP 5. Je nabízený pod licencemi GNU GPL a Nette. Tento framework se zaměřuje na eliminování bezpečnostních rizik a podporu znuvupoužitelnosti kódu. Mezi nesporné výhody použití Nette patří dobré ladící nástroje a aktivní komunita uživatelů, která přispívá k rychlému vyřešení dotazů ke vzniklým problémům. Další výhodou je logování chyb při nasazení do provozu, což je vhodné pro pozdější odstranění problémů, které se projeví po nějakém čase.

5.1.6 MySQL

MySQL [9] je volně šiřitelná multiplatformní databáze, díky čemuž je nejvhodnější pro použití v této práci. Komunikace s databází probíhá pomocí standardizovaného dotazovacího jazyka SQL

používaného pro dotazování nad drtivou většinou databází. V rámci MySQL se používá stejně jako u ostatních databází dialekt tohoto jazyka s určitými rozšířeními.

Databáze jsou využívány k ukládání dynamicky se měnících dat. Databáze je většinou šifrovaná a pro přístup k ní je potřeba vlastnit certifikát, nebo alespoň znát přihlašovací údaje, což zajišťuje bezpečnost uložených i v případě napadení serveru provozovatele. Dále databáze umožňují oddělení dat programu od uživatelských, což do vzniku databází neplatilo.

V popisovaném programu je databáze použita pro uložení časů pro automatické spínače a pro zaznamenání času změn stavů pro všechna zařízení.

5.1.7 Jazyk C a VHDL pro FITkit

FITkit je rozdělen na dvě programovatelné části. První částí je řídící jednotka MCU (Multipoint Control Unit) a druhou hradlové pole FPGA (Field Programmable Gate Array).

Řídící jednotka MCU je programována pomocí jazyka C99 překládaného pomocí GNU překladače. Jazyk C99 [12] je přenositelný procedurální programovací jazyk vyvinutý pro potřeby operačního systému linux. Tento jazyk je nízkoúrovňový a tím pádem vhodný pro programování operačních systému, čehož je využito při programování MCU. Jazyk C99 dovoluje modulární programování, což je výhodné při práci v týmu a pro zvýšení přehlednosti výsledného kódu.

Hradlové pole FPGA se programuje pomoví jazyka VHDL (VHSIC Hardware Description Language, kde VHSIC znamená Very-High-Speed Integrated Circuit). Zde se nejedná o programování jako takové ale spíše o popis hardware, což usnadňuje provádění změn v systému, kdy stačí přepsat požadovanou část v kódu a přeprogramovat FPGA. Jazyk VHDL se používá pro návrh a simulaci digitálních integrovaných obvodů. VHDL byl standardizován v roce 1987 a od revize v roce 1997 je použitelný také pro návrh analogových obvodů. Je to typový jazyk s prostředky pro popis paralelismu, konektivity a explicitního vyjádření času. VHDL disponuje více styly popisu architektury, například behaviorální styl, kdy si návrhář řídí architekturu návrhu sám.

5.2 Bezpečnost

Důležitou součástí práce je bezpečnost. Bezpečnost se týká přenášených dat a autentizace uživatele. V případě nepřítomnosti těchto věcí je možné zneužít systém, sloužící k zabezpečení, ke ztrátě bezpečnosti objektu.

Jedním z použitých bezpečnostních prvků je využití šifrovaného HTTP protokolu (HTTPS) [18]. Komunikace na tomto protokolu funguje mezi serverem, na kterém běží webová aplikace a vzdáleně připojovaným klientem a také mezi klientem a IP kamerou. Dalším bezpečnostním prvkem je použití firewallu na směrovači, kterým je celý systém připojen k internetu. Firewall musí blokovat veškerou nevyžádanou komunikaci, díky které by mohlo dojít k odposlechu dat v místní síti a poté k jejich zneužití, či podvržení daty upravenými. Přístup k celému systému je řešen autentizací pomocí uživatelského hesla a jména, což při vhodně zvolené kombinaci odolá většině útoků na systém.
Do budoucna by bylo vhodné šifrovat i přenos mezi řídící jednotkou a serverem. Jelikož v této komunikaci slouží řídící jednotka, realizována na FITkitu, jako server, je implementace HTTPS serveru nad rámec rozsahu práce. K odposlechu nebo podvržení této komunikace může dojít pouze při fyzickém narušení sítě. K fyzickému narušení sítě může dojít pouze po vstupu do objektu, na což systém dokáže zareagovat a upozornit uživatele. Po narušení objektu je dobré systém otestovat a zkontrolovat, zda narušitel neohrozil bezpečnost systému a v případě že ano, tyto problémy odstranit.

5.3 Realizace na FITkitu

FITkit realizuje v systému přístupový modul mezi připojenými zařízeními a webovým serverem. Realizace na FITkitu je složena ze dvou částí a to části microkontroléru MCU, a části FPGA.
FPGA část se stará o komunikaci s připojenými zařízeními. Zařízení jsou připojena na vstupně-výstupním rozhraní JP10 (toto rozhraní je připojeno pouze k FPGA, jinak by FPGA nebylo v práci vůbec potřeba).

Kvůli přehlednosti jsem rozdělil připojené porty na tři sektory. Na prvním sektoru, který je na portech jedna až dvacet, je připojen ethernetový modul. Na druhém sektoru, který je na portech dvacettři až třicetsedm jsou připojena výstupní zařízení a ve třetím sektoru, který je na portu čtyřicet až padesátdva jsou připojena vstupní zařízení.

Ethernetový modul má vlastní komunikační protokol s FITkitem. Tento problém není obsahem mojí práce, proto o tomto problému více psát nebudu.

Výstupní zařízení jsou spínána nastavením daného portu do logické jedničky a na jejich výstupu je referenční napětí, kterým je sepnuto relé. Relé je připojeno na zařízení, které je potřeba sepnout.

Vstupní zařízení jsou dvojího typu. Prvním typem jsou dvoustavové a druhým jsou spínací zařízení. Kvůli spínacím zařízením, která nepodporují přivedení země v jednom stavu a referenčního napětí v druhém, je potřeba v části FPGA pracovat s referenčním napětím jako s logickou nulou a se zemí jako s logickou jedničkou. Tomuto chování je možno předejít pomocí přepínacího relé, které ale potřebuje samostatný napájecí obvod, což by ve velkém objektu mohlo znamenat i desítky napájených obvodů navíc.

Nyní známe v FPGA stav připojených čidel a umíme změnit stav výstupních zařízení. Dále je potřeba přenést stav z a do MCU. K tomu jsem ve svém projektu využil sběrnici SPI. K realizaci komunikace pomocí SPI je v FPGA připravena komponenta, ke které je nutné pouze připojit požadované signály. V MCU jsou pro tuto komunikaci k dispozici potřební knihovny. Každá SPI komunikace probíhá na jiné, předem dané, adrese v paměti.

V mém případě sdílí vždy jedno vstupní a jedno výstupní zařízení jednu adresu. Pro komunikaci se používá funkce FPGA_SPI_RW_A8_D8. Pro zjištění stavu vstupních čidel se tato funkce volá s parametrem SPI_FPGA_ENABLE_READ, dalším parametrem je adresa dané komunikace a posledním parametrem je právě čtená hodnota, a to buď logická nula, nebo jednička. Pro nastavení hodnoty je funkce volána s parametrem SPI_FPGA_ENABLE_WRITE, s dalším parametrem, kterým je adresa komunikace a posledním parametrem, kterým je předávaná hodnota.
Další důležitou částí je realizace komunikace mezi FITkitem a webovou aplikací. Komunikace je realizována pomocí protokolu TCP na portu 80. Na FITkitu je vytvořen soket a zaregistrován handler na port 80, poté je přepnut právě vytvořený server do stavu naslouchání.

Předávání dat je prováděno pomocí HTTP protokolu. Na FITkitu běží jednoduchý webový server, na který se webová aplikace dotazuje pomocí standardizovaného dotazu GET. Dotazy mají přesně daný tvar, podle kterého se rozhodne, o jaké jde zařízení a vrátí jeho aktuální stav, nebo u výstupních jeho stav nastaví.

Aktuální stav všech zařízení je uchováván v MCU. A při změně stavu v FPGA, nebo při požadavku na přepnutí spínače, je stav změněn.

5.4 Realizace webové aplikace

Popis realizace webové aplikace jsem rozdělil do dvou částí. V první části se zabývám popisem komunikace a v druhé jsem se zaměřil na popis časového spouštění, termostatu a seznamu stavů zařízení.

5.4.1 Popis komunikace

Webová aplikace se stará o zobrazování stavu všech čidel a senzorů. Stará se také o předání změny stavu čidel vyžadovaných uživatelem. O komunikaci a zobrazení se stará JavaScriptová část.
O komunikaci směrem k FITkitu se stará funkce *setState*, která očekává jako povinný parametr identifikační číslo zařízení, které má být přepnuto. Tato funkce zpracuje přijatý parametr a pošle dotaz GET na IP adresu FITkitu ve tvaru

$$http://IPadresaFITkitu]:80/index.html?device=[identifikačníčíslozařízení]$$

Na tento dotaz očekává jako odpověď stav zařízení, který se udržuje v MCU FITkitu. Odpověď je buď logická nula nebo logická jednička. Funkce podle vrácené hodnoty nastaví plochy přiřazené přepnutému spínači. Barva oznamovací oblasti je závislá na druhu zařízení, protože například sepnutí závlahy se zobrazí zeleně, když je závlaha v provozu. Na rozdíl od toho se garážová vrata zobrazují červeně, pokud dojde k sepnutí a otevření.

Komunikace z FITkitu směrem k uživateli je také iniciována JavasScriptem a to v pravidelných intervalech (interval tři vteřiny). Komunikace probíhá podobně jako v opačném případě pomocí dotazu GET, pouze tvar je odlišný. Dotaz má tvar

$$http://[IPadresaFITkitu]:80/get.html$$

Tento dotaz je pro všechna čidla a senzory společný z důvodů minimalizace síťové komunikace a v odpovědi očekává stavy všech zařízení. Vrácený řetězec je rozdělen na stavy jednotlivých zařízení a tyto stavy jsou nastaveny jednotlivým vizualizačním blokům. Na konci řetězce jsou posílány i stavy vzdáleně spínaných zařízení kvůli počáteční inicializaci, jelikož stav zařízení se udržuje v MCU FITkitu a navíc může dojít k ručnímu přepnutí bez použití vzdáleného systému.

Poslední komunikací je dotaz na aktuální teplotu. Dotaz je ve tvaru

$$http://[IPadresaFITkitu]:80/teplota.html$$

a očekává v odpovědi teplotu ve tvaru desetinného čísla. Tento dotaz je prováděn v intervalu tří sekund společně s předchozím. K přijaté teplotě je přidán znak stupňů celsia pro zobrazení.

5.4.2 Časové spouštění, termostat a seznam změn stavů

Nastavování časovače spouštění je řešeno pomocí formuláře s potvrzovacím tlačítkem. Nejprve je nutné vybrat zařízení ze seznamu nabízených zařízení. Po vybrání zařízení se v políčcích pro zadávání dat zobrazí aktuálně nastavená data. Časovač očekává zadání spínacího a vypínacího času ve tvaru hh:mm:ss. Pokud je potřeba časovač vypnout, je nutné nechat obě zadávací pole prázdná. Tvar zadávaných dat je kontrolován pomocí regulárního výrazu. Časovač porovnává nastavené časy s aktuální časem nastaveným na spuštěném počítači (serveru).

Nastavení termostatu je řešeno také pomocí formuláře s potvrzovacím tlačítkem a příliš se neliší od časovače spouštění. Nejprve je nutné vybrat zařízení ze seznamu, čímž dojde k zobrazení aktuálních dat, stejně jako u předchozího časovače. Termostat očekává zadání spínací a vypínací teploty ve tvaru celého čísla a v rozmezí od mínus 99 do plus 99. Pokud je potřeba termostat vypnout, je potřeba nechat obě pole prázdná. Termostat porovnává zadané teploty s aktuální teplotou získanou z řídící jednotky.

Seznam změn stavů je také realizován pomocí formuláře. Tento formulář obsahuje dvě vstupní pole pro zadání data, potvrzovací tlačítko a seznam vybraných záznamů. Ve výchozím stavu je nastaveno rozmezí na poslední den od aktuálního data. Do formuláře se zadává datum od kdy a do kdy mají být záznamy zobrazeny. Datum musí být zadáváno ve tvaru rrrr-mm-dd. Pro lepší orientaci a zadávání u každého políčka vyskočí okénko s kalendářem, ze kterého se po klinutí na vybrané datum políčko vyplní. Po zmáčknutí tlačítka vyhledat se v seznamu zobrazí záznamy z vybraného rozmezí.

5.5 Popis databázové struktury

Databáze obsahuje dvě tabulky. První tabulka je tabulka zařízení – devices, druhá je tabulka záznamů změn stavů zařízení – log. Tabulka devices má šest sloupců. První je device_id, což je jednoznačný identifikátor zařízení, další je jméno zařízení name sloužící pro lepší orientaci. Třetí a čtvrtý sloupec tvoří pár pro časovač. Jsou to sloupec start_time reprezentují čas od a sloupec stop_time reprezentující čas do. Poslední dva sloupce jsou taky párové a slouží pro termostat. Sloupec low_temp reprezentuje spodní hranici spuštění a max_temp reprezentuje horní hranici spuštění. Druhá tabulka má název log. Tato tabulka má čtyři sloupce, z toho jeden obsahuje jako cizí klíč identifikátor zařízení device_id z tabulky devices. Sloupec id je jednoznačný identifikátor vloženého stavu. Sloupec state obsahuje stav zařízení v čase záznamu. Poslední je *date* obsahující datum vložení záznamu.

5.6 Vizuální reprezentace systému

Systém je vizuálně reprezentován jako mapa sledovaného objektu s bloky sloužícími pro oznamování stavu čidel a senzorů a bloky reprezentujícími tlačítka pro přepínání spínačů.

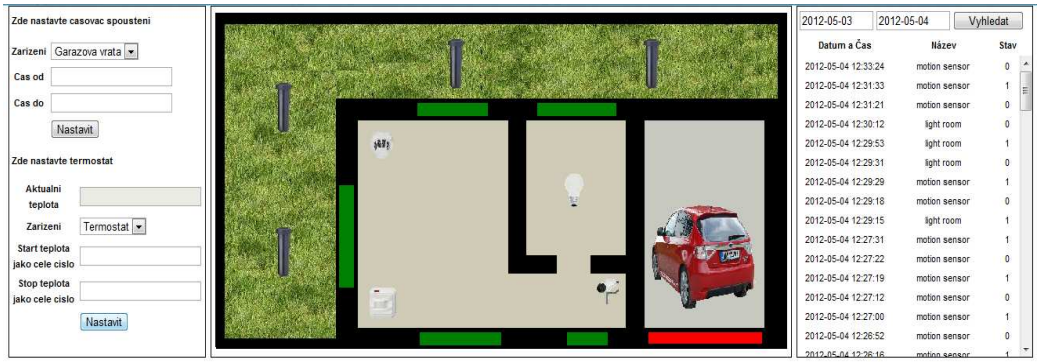
Vizualizace čidel a senzorů je rozdělena do dvou částí, a to na bloky, které mění barvu podle stavu čidla a čidla měnící obrázek. Bloky měnící barvu se využívají pro zobrazení signalizace zabezpečení přístupových cest. V případě zavření dveří, respektive oken, je spínač nebo magnetický kontakt sepnut, což je zobrazeno zelenou barvou. Při otevření dveří, respektive okna dojde k rozpojení spínače nebo magnetického kontaktu, což je zobrazeno kontrastně oproti klidovému stavu červenou barvou. Změna zobrazení změny stavu čidla se děje nejpozději do tří vteřin od fyzické změny stavu. Barvu poplachu jsem zvolil záměrně červenou, která vyvolává stav nebezpečí, proto i ve druhé variantě, která využívá k vizualizaci obrázek daného senzoru, převládá při stavu ohrožení červená barva. Zvolení zobrazení pomocí obrázků je kvůli lepší orientaci. Pokud by byly na plánku pouze barevné bloky, mohlo by dojít z nevědomosti uživatele o jaké zařízení se jedná, a tím způsobit dezinformaci, například v případě záměny pohybového čidla s požárním detektorem. V případě takové záměny by mohlo dojít ke zbytečnému kontaktování složek záchranného systému.

Druhým typem jsou bloky reagující na stisk. Tyto bloky jsou také vizualizovány obrázky nebo bloky přebarvujícími se podle stavu. Zde u přebarvování využívám stejné barvy, ale výsledná reprezentace se v podstatě chová opačně, a to zelená při sepnutém stavu a červená při vypnutém stavu. U obrázků jsem zvolil barvy podle druhu spínaného zařízení, například žárovka má ve vypnutém stavu pouze žlutý obrys a ve stavu sepnuto je vybarvena žlutě.

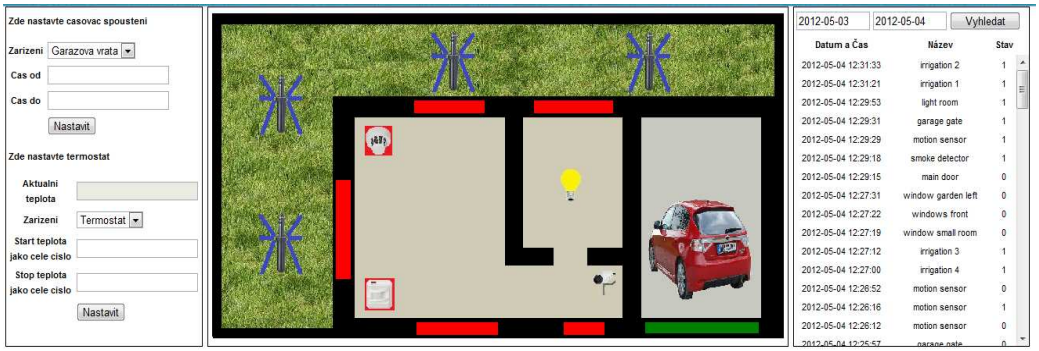
Posledním použitým vizualizačním prvkem je tlačítko pro spuštění videa s IP kamery.

Blok s mapou je obalen formuláři. Na levé straně jsou formuláře pro nastavení časového spouštění a nastavení termostatu. Na pravé straně je formulář pro zobrazení historie stavů zařízení, který dovoluje scrollování zobrazenými záznamy, čímž se předchází narušení designu stránek.

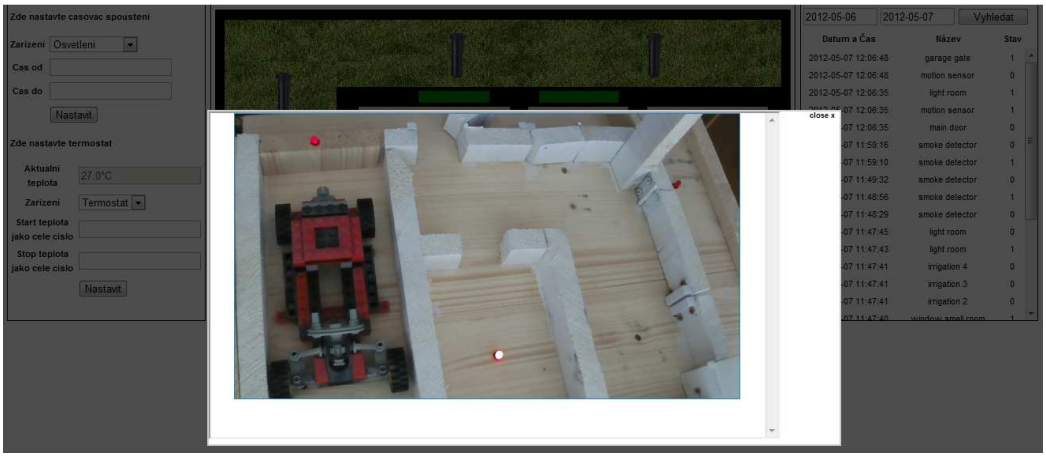
Pro zobrazení videa s IP kamery je použito vyskakovací okno umístěné na středu obrazovky. Obraz z kamery je zobrazen v levném horním rohu tohoto okna a roztažen podle zvolené velikosti videa a dále kamera dovoluje nastavit světlost, kontrast, saturaci a další.



Obrázek 6: Screenshot systému v klidovém stavu.



Obrázek 7: Screenshot systému ve stavu alarmu a použití všech zařízení.



Obrázek 8: Screenshot systému s obrazem z webkamery.

5.7 Zprovoznění a zavedení systému

Zavedení systému do reálného provozu vyžadovalo v první řadě analýzu rozmístění všech zařízení. Dále bylo nutno navrhnout plánek budovy rozmístěním všech periferií. Také bylo potřeba připojit řídící jednotku (FITkit) pomocí ethernetového kabelu do stejné sítě jako je připojen počítač, na němž

běží webový server. Dále bylo nutné nastavit v souboru Default.map.latte stejnou IP adresu jako má FITkit (v ideálním případě tato operace není nutná, ale FITkit někdy použije adresu s DHCP a ne ručně nastavenou adresu). FITkit bylo dále potřeba připojit ke zdroji elektrické energie, což je možné pomocí externího 5 voltového adaptéru nebo pomocí USB kabelu. V případě připojení externího zdroje je potřeba rozpojit propojky J8 a J9. Naopak v případě připojení přes USB je potřeba pokaždé připojit FITkit v terminálu, například v nástroji QDevKit. Poté bylo nutné připojit zařízení na port k tomu určený. Dvoustavová vstupní zařízení se připojují jedním kontaktem k portu a druhým k zemi. U dvou stavových zařízení bylo potřeba připojit výstupním kontaktem na daný port.

5.8 Použité zařízení a technologie

K zabezpečení vstupních cest jsem využil magnetický kontakt značky Longhorn. Tento spínač je dvoudrátový zalitý v plastovém pouzdře umístěný na pevné části okna, který je sepnut druhou částí obsahující magnet zalitou v plastovém pouzdře a je připevněný na pohybující se části okna nebo dveří. Další součástí systému je pohybový spínač značky Longhorn, typu PDS-901A. Zařízení monitorující stav kuchyně je detektor kouře Longhorn gas detektor LH-87. Pro vizuální kontrolu slouží IP kamera značky airCam připojená do stejné sítě jako FITkit a webový server. Tato kamera má zabudovaný vlastní webový server, který po HTTP dotazu vyvolaným uživatelem vysílá požadovaný obraz. FITkit je použitý ve verzi 2.0, kvůli připojení ethernetového modu, který vyžaduje použití tohoto modulu. Jako zařízení pro síťovou komunikaci jsem využil WIRELESS N 150 HOME ROUTER od firmy DLINK. Toto zařízení není použito kvůli nějaké speciální funkčnosti, ale kvůli dostupnosti, kdy nebyla nutná žádná další investice. V případě nasazení do reálného systému je možné využít již používané zařízené, kterým v dnešní době disponuje většina domácností nebo kanceláří.

Pro zapínání a vypínání zařízení by bylo potřeba využít elektromagnetické relé. Návrh počítá s využitím relé od značky Sun Hold typu TNA-2C-0505L. Dané relé je spínané referenčním napětím FITkitu a dokáže spínat obvod, kterým protéká střídavý proud 5 A a napětí má 250 V. Ve vyrobeném modelu nebylo relé využito a z důvodů bezpečnosti obsluhy modelu je spínání simulováno pomocí LED diod.

Všechna použitá zařízení jsou detailně popsána v kapitole 3, kde je možné zjistit přesné zapojení, referenční napětí a veškeré potřebné parametry.

5.9 Popis komunikačního protokolu

Komunikační protokol je navržen jednostranně kvůli problému s častým navazováním a ukončováním spojení. Dotaz je vždy realizován ze strany webové aplikace a FITkit pouze odpovídá. Dotazy jsou rozděleny do dvou druhů, a to dotazy pro výstupní zařízení a dotazy pro vstupní zařízení.

Pro zjištění stavu vstupních zařízení se webová aplikace dotazuje každé tři sekundy. Dotaz musí být ve tvaru

$$http://[IPadresaFitkitu]:80/get.html$$

Odpověď je tvaru *[aktuální stav prvního zařízení];[aktuální stav druhého zařízení]; … ;[aktuální stav posledního zařízení]*. Odpověď je poté rozparsována pomocí funkce *split*.

Pro nastavení stavu výstupních zařízení se používá dotaz ve tvaru

$$http://[IPadresaFitkitu]:80/index.html?device=x$$

X je číslo zařízení. Jako odpověď na tento dotaz je jedno číslo a to stav zařízené reprezentující logickou nulu, nebo logickou jedničkou.

Dalším dotazem je dotaz

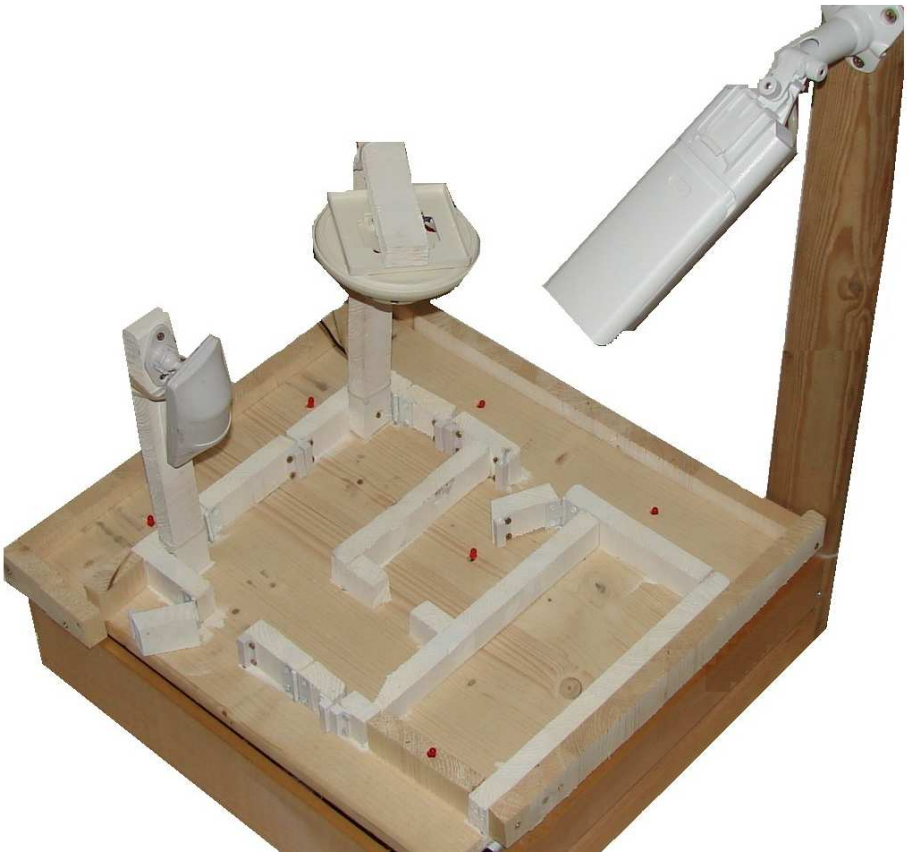
$$http://[IPadresaFitkitu]:80/teplota.html$$

Tento dotaz vrátí aktuální teplotu teplotního čidla FITkitu.

5.10 Popis modelu vytvořeného systému

Celý systém je umístěn na dřevěném boxu, ve kterém je schována veškerá kabeláž, řídící jednotka a síťový prvek. Síťový prvek se skládá z routeru, switche a wifi přístupového bodu. Tento prvek zajišťuje veškerou síťovou komunikaci potřebnou pro správný chod systému. Na horní straně boxu jsou umístěny kvádry imitující stěny a dveře spojené pomocí pantů. Na stěnách jsou umístěny spínací magnetické kontakty připojené kabelem k řídící jednotce a na dveřích jsou umístěny volné části těchto kontaktů. Senzory a kamera jsou umístěny na stožárech připevněných na horní část modelu. Výstupní vzdáleně ovládané zařízení jsou v modelu reprezentována LED diodami. Fyzické umístění zařízení odpovídá umístění na mapě objektu webové aplikace.

Fyzické zapojení systému je popsáno v příloze schéma zapojení vyrobeného modelu. V tomto schématu je popsáno jak zapojení síťové komunikace, symbolizované čerchovanou čarou, tak zapojení komunikace mezi sběrnicí a zařízeními je symbolizováno plnou čarou. Zařízení jsou na schématu symbolizována bloky s názvy popisovaného zařízení. U senzorů, které potřebují samostatné napájení, je symbolicky naznačeno napájení a připojení do systému k danému portu. Připojení zařízení, která vyžadují specifické připojení, je přesně popsáno v kapitole 3. Připojení magnetických kontaktů a LED diod není popsáno pomocí speciálních schémat. Připojení magnetického kontaktu lze vyčíst z celkového schématu. U LED diod je připojení obdobné, pouze je potřeba dodržet polaritu naznačenou ve schématu.



Obrázek 9: Foto modelu systému.

6 Popis zdrojových kódů

V této kapitole bude popsána souborová struktura a umístění funkcí s jejich významem. Kapitola je rozdělena na tři části. V první části je popsána centrální řídící jednotka a v další části webový server.

6.1 Řídící jednotka

Popis řídící jednotky je dále rozdělen na dvě samostatné části. První část se zabývá popisem jádra FITkitu a druhá popisem FPGA.

6.1.1 MCU jádro FITkitu

Jádro je popsáno pouze v jednom souboru *bp-main.c*. I když jazyk C dovoluje modularizaci, tak jsem ji z důvodu jednoduchosti a ne příliš velkému rozsahu nevyužil.

Na začátku souboru jsou připojeny potřebné knihovny pro práci s FITkitem a s jeho moduly. Následuje definice adres pro komunikaci SPI a globální proměnné, ve kterých jsou uloženy stavy připojených zařízení. První funkce *print_user_help* slouží pro vypsání uživatelské nápovědy. Nápovědu lze zobrazit v konzole pro připojení FITkitu k počítači, která v ideální sestavě systému nebude použita, ale lze využít při sestavování výsledného produktu. Funkce decode_user_cmd není využita, ale překladač ji vyžaduje. Další je funkce fuga_initialized, ve které se inicializují periferie po restartu. Následující get_sensor se stará o komunikaci s FPGA a to ve smyslu získávání stavu zařízení, s nimiž komunikuje právě FPGA a vrací po dotazu stav ve tvaru jedna nebo nula. Handler TCP_APP_HANDLER je volán vždy při příchozí komunikací TCP a je základem řídící jednotky. Handler je rozvětven pomocí podmíněného příkazu if-else, kde se v podmínce porovnávají data přijaté metodou GET http protokolu. Dotazy jsou rozděleny do tří různých typů. První je dotaz pro dálkově ovládané zařízení, kde podle přijatého identifikátoru zařízení se přepne jeho stav a odešle odpověď o úspěšném přepnutí. Další podmínka je splněna, když se aplikace pravidelně dotazuje na aktuální teplotu a pokud vše proběhne v pořádku, je odeslána odpověď obsahující požadovanou teplotu. Posledním dotazem s kladnou odpovědí je dotaz pro získání stavů všech připojených zařízení. Úplně poslední podmínka je při normálním běhu aplikace nesplnitelná, ale zůstává pro účely sestavování systému. V hlavní funkci programu main jsou pouze počáteční inicializace a výpis na komunikační konzole využívaný pro účely zapojení a testování.

6.1.2 Popis hardwaru v FPGA

Popis hardware je proveden pomocí jazyka VHDL propojením signálů. Propojením signálů je simulováno zapojení fyzických součástek.

Celý popis začíná zvolením používaných knihoven a použité architektury. V první části je potřeba definovat signály pro propojující zařízení. Jako první jsou umístěny signály pro ethernetový modul a za nimi následují signály pro komunikaci s čidly a pro komunikaci pře SPI. Následuje definice komponenty pro SPI komunikaci ve tvaru, který očekává knihovny obstarávající tuto komunikaci. Za počáteční definicí je umístěn SPI dekodér a kontrolní proces pro ethernetový modul, které jsem převzal ze vzorové aplikace vytvořené k tomuto modulu. Následují SPI dekodéry pro komunikaci s MCU řídící jednotkou. Každý dekodér má nastavenou adresu stejnou jako je v MCU u daného zařízení a dále jsou signály DATA_IN a DATA_OUT připojeny na porty spojené s obsluhou

vybraného zařízení. Počet dekodérů odpovídá maximálnímu počtu vstupních nebo výstupních zařízení, jelikož každý dekodér zvládne obsloužit jak vstupní tak výstupní zařízení dohromady. Následuje proces spouštěný s každým taktem, který nastavuje hodnotu na výstupním portu. To probíhá tak, že pokud je uskutečněna komunikace z MCU, tak se přičte jedna k aktuálnímu stavu, což u jednobitového signálu způsobí inverzi. V dalším bloku dochází k přidružení signálů k příslušnému portu na výstupní sběrnici, a to pomocí podmíněného příkazu if-else. Poslední částí je mapování portů pro komunikaci s ethernetovým modulem připojeným na stejnou sběrnici jako zařízení.

6.2 Webový server

K vývoji aplikace realizující webový server jsem využil Nette Framework, který vyžaduje rozdělení do samostatných souborů. Tyto soubory propojuje framework podle názvů, které musí mít přesně daný tvar a koncovku. Díky použití tohoto frameworku je možné jednoduše oddělit od sebe části prováděné v php od částí html a následné je pomocí syntaxe frameworku jednoduše propojit. Další výhodou je možnost využití objektového programování skrze oddělené soubory.

Stěžejním souborem je soubor *Default.map.latte*, ve kterém je kostra následně dynamicky vygenerovaného html souboru. Tento soubor má syntaxi stejnou jako html soubor, která je obohacena o prvky nette a samozřejmě mohou být vkládány JavaScriptové části. Na začátku tohoto souboru je popsáno umístění prvků hlavní obrazovky, které jsou popsány v souboru layout.css (struktura bude popsána později). Tyto prvky jsou řazeny chronologicky tak, jak jsou umístěny na obrazovce a to od levého horního rohu. Prvním prvkem je tabulka, obsahující formuláře pro nastavení časovače spouštění a termostatu. V této části kódu stojí za poznamenání vložení aktuálně nastavených hodnot podle vybraného zařízení v comboboxu, kde pokud není nic nastaveno, zůstane políčko prázdné.

V další části je umístěna mapa objektu. Zde stojí za zmínku využití syntaxe nette ke změně barvy nebo obrázku podle stavu uloženého v databázi. Díky nette zde lze jednoduše využít podmínku if-else, což by se v případě samotného html bylo mnohem složitější. V poslední části je vložen logbox, který podle vybraného data zobrazuje změny stavů všech zařízení v zadaném období. Datum je nastaveno ve výchozím stavu na jeden den dozadu z důvodů přehlednosti výpisu. Pro zadání data jsem využil komponentu *datapicker* umožňující vybrání požadované hodnoty z přehledné tabulky, čímž je dosaženo zadání přesného tvaru data. Komponenta *datapicker* je převzatá a pouze upravená pro potřeby aplikace.

V kódu následuje komponenta pro zobrazení obrazu z IP kamery. K tomuto zobrazení využívám vyskakovací okno, které je také převzaté a upravené pro vlastní potřebu, jelikož nemá smysl vymýšlet již vymyšlené.

Ve zbytku souboru jsou již části JavaScriptu obsahující funkce, které pracují takzvaně ajaxově (na pozadí). První funkce je *setState(buttonId)*, která je volána po stisku tlačítek v mapě objektu, a je jí předáváno identifikační číslo zařízení. Zmíněná funkce odešle dotaz *GET* do řídící jednotky, což vyvolá změnu stavu zařízení s odpovídajícím identifikátorem. Druhá je funkce *getState()* volaná pravidelně každé tři vteřiny a dotazuje se také pomocí dotazu *GET* řídící jednotky na aktuální stav všech připojených ovládacích i ovládaných prvků. Řídící jednotka odešle nazpět řetězec obsahující aktuální stavy. Na začátku se zavolá php funkce, která vloží do databáze stavů ty stavy, které se za poslední tři sekundy změnily. V příchozím řetězci se shoduje pořadí čísla s identifikátorem, a tím pádem je možné využít vestavěnou funkci JavaScriptu *split*, která stavy oddělené středníkem vloží do pole. Naplněné pole je dále procházeno po prvku a podle hodnoty je zobrazen požadovaný stav. Jedinou hodnotou, pro kterou je prováděn samostatný dotaz, je teplota.

V poslední části, která provede refresh stránky vždy, když jsou změněna data v databázi, je část starající se o automatické časové spínání a termostat. Termostat pracuje s aktuální teplotou,

kterou porovnává s maximální a minimální teplotou zvolenou uživatelem a podle toho zapíná a vypíná topení či klimatizaci, podle toho, co uživatel požaduje. Časovač porovnává zvolený spouštěcí i vypínací stav s aktuálním časem na serveru. První se u časovače porovnává, který ze dvou nastavených časů je větší. Pokud je čas zapnutí menší než čas vypnutí, probíhá interval spuštění v rámci jednoho dne a zařízení je sepnuto, pokud je aktuální čas větší než čas sepnutí a menší než čas vypnutí. V druhém případě je ohlídán přechod do dalšího dne, kde je potřeba otočit intervaly. Další soubory typu nette s koncovkou .latte jsou Default.default.latte, @layout.latte a Default.insert.latte. První soubor obsahuje pouze umístění prvku pro přihlášení a vazbu na php funkci userLogin, která bude popsána dále. V druhém souboru je definována hlavička výsledného html souboru se všemi připojovanými skripty a poslední soubor je připraven pro rozšíření při připojení více řídících jednotek pro příchozí dotazy.

Nette dovoluje oddělit části kódu v php úplně do samostatných souborů. Dále jsou rozděleny na presentery a medely. První soubor v presenterové části je soubor *DefaultPresenter*.php. V tomto souboru jsou umístěny funkce komunikující s databází, funkce pro vytváření formulářů a render předávající proměnné do souboru *Default.map.latte*. V redneru je navíc přenastaveno výchozí datum na rozmezí aktuální mínus jeden den a tak aby při zadání stejných dat byl zobrazen jeden den. Následují funkce nazvané handle sloužící pro komunikaci s databázovými dotazy. V poslední části jsou umístěny funkce pro vytvoření formulářů a funkce pro jejich obsluhu. První je formulář pro vytvoření logboxu. Jeho obsluha je velmi jednoduchá a to pouze zobrazit záznamy v databázi pro zvolené rozmezí. Další formulář slouží pro uložení dat pro časové spouštění. Ten se skládá ze dvou textových polí a jednoho potvrzovacího tlačítka. Textová pole musí být v požadovaném tvaru, což zajišťuje podmínka daná regulárním výrazem. Požadovaný tvar je *hh:mm:ss* nebo prázdné pole, což znamená nespuštěno. Posledním formulářem je termostat, který také obsahuje dvě textová pole a jedno potvrzovací tlačítko. Zde se vstupní data testují, jestli patří mezi čísla a spadají do rozsahu -99°C až +99°C. Další podmínkou je to, že minimální teplota musí být nižší než maximální teplota. Další soubor v presenterové části je *BasePresenter.php* starající se o přihlašování a správu uživatelů. Tento soubor je z velké většiny převzatý z příkladů v dokumentaci k nette a je upraven podle požadavků aplikace.

Další oddělené php soubory jsou *Query.php*, *DBtype.php*, *UserSet.php* a *Users.php*. Tyto soubory patří do modelové části. První dva jsou určeny pro práci s databází. V souboru DBtype je umístěna pomocná třída pro práci s databází. V Query jsou uloženy databázové dotazy. Funkce insertData obsahuje dotaz, který vloží do databáze řádek při změně stavu zařízení a využívá následující pomocnou funkci checkState. Další je funkce getLog obsahující dotaz na tabulku stavů zařízení. Tento dotaz vybere všechny odpovídající řádky tabulky podle zadaného počátečního a koncového data. Dotaz využívá spojení dvou tabulek a díky syntaxi jazyka dibi je možné využít podmíněný výraz přímo v dotazu, pro porovnání s vybraným datem. Následuje funkce getLastState vrátí poslední aktuální stav zařízení. Další funkce updateTimeRange a updateThermoRegulation upraví časy respektive teploty spouštění v tabulce zařízení. Poslední dvě funkce jsou getDevices a getTemperatures. První z nich vrací obsah celé tabulky devices a druhá vrací maximální a minimální teplotu pro sepnutí, taky z tabulky device. Soubor UserSet.php slouží k nastavení uživatelských rolí. V aplikaci je uživatelská přístupová role pouze jedna, ale komponenta nette pro práci s uživateli tento soubor vyžaduje a v případě rozšíření aplikace do budoucna je možné role přidat. Poslední model je soubor Users.php starající se o autentifikaci uživatelů, zjištění a přiřazení jeho role.

Rozvržení prvků na stránce je popsáno v souboru layout.css. Prvky jsou rozděleny do bloků podle významu a použití. Každý prvek má nastavenu velikost, pozici a pozadí. U prvků, které mění barvu podle stavu zařízení, je přebarvení prováděno pomocí tříd on a off. Třídy obsahují barvu pozadí a jsou připojeny k prvkům podmíněným výrazem. Tam, kde nestačí přebarvení, vytvářím dva podobné prvky lišící se obrázkem pozadí a měním jejich viditelnost. Speciálním případem je

zobrazení obrazu z kamery. K tomuto je použito vyskakovací okno. Tohle vyskakovací okno jsem využil již dříve vytvořené mým kamarádem Markem Skopalem.

Doposud nepopsané soubory jsou volně dostupné JavaScriptové knihovny a konfigurační soubory pro nette. Jediný JavaScript, který není volně dostupný, je popup.js vytvořený již zmiňovaným kamarádem a mnou upravený.

7 Testování a možná rozšíření

Testování systému je nedílnou součástí vývoje. Testy je nutné provádět při každé změně ve funkčnosti programu. Pokud nejsou testy prováděny pravidelně při každé změně, může dojít k tak závažné chybě v systému, která nemusí mít návratu, nebo minimálně může zpomalit vývoj, i o několik desítek procent celkového času.

7.1 Testování vytvořeného systému

Testování probíhalo ve více fázích. V první fázi jsem vytvořil část softwaru pro řídící jednotku, na kterém jsem otestoval funkčnost všech vybraných čidel a spínačů. Pro potřebu tohoto testování jsem zprovoznil pouze jeden port, na kterém jsem všechna zařízení střídal a testoval jejich funkčnost. Následovala zprovoznění a otestování všech ostatních potřebných portů pomocí mikrospínače, který jsem zvolil jako nejvhodnější pro testování. Díky testování jsem odhalil spojení některých portů dohromady a vyřadil jsem nepoužitelné. Po otestování portů jsem připojil ethernetový modul a za pomoci přeprogramovaných aplikací jej zprovoznil pro účely mé aplikace. Při testování jsem zjistil, že původní plán komunikovat obousměrně nebude fungovat, a proto jsem zvolil komunikaci, kdy je FITkit dotazován webovou aplikací. Po zprovoznění modulu přišlo na řadu testování komunikace mezi FITkitem a webovou aplikací. Z testování vyplynulo, že je nutné upravit hlavičku http protokolu, tak aby jí rozuměla webová aplikace. Problém byl také v nutnosti nekomunikovat jenom v rámci aplikace, ale komunikovat s jiným serverem. Tento problém byl odstraněn změnou podpůrného skriptu nette frameworku. Po vyřešení problému v komunikaci byl již systém připraven na implementaci do modelu.

Další fází testování bylo otestování vyrobeného modelu, který byl již v téhle fázi osazen senzory, spínači, diodami a kamerou. Po propojení modelu s řídící jednotkou přišlo na řadu kompletní testování celého modelu. Testování na vytvořeném modelu probíhalo více než týden. K testování jsem vybral některé své přátele a rodinu. S testery jsem formou pohovoru provedl zhodnocení aplikace. Z výsledků hodnocení aplikace jsem navrhl drobné změny rozložení prvků a doplnil přísnější kontrolu vstupních dat, jelikož došlo k několika vložením nesprávného tvaru dat. Po veškerých úpravách byla provedena poslední série testů, která neukázala žádný problém a potvrdila správnou funkčnost systému.

Aplikace byla vyvíjena a odladěna na webovém serveru IIS 6.1 a za použití webového prohlížeče google chrome od verze 15 výše. Testování probíhalo na stejném serveru a na různých webových prohlížečích dodržujících standardy.

7.2 Možná rozšíření

K systému, který obsahuje jeden FITkit, je možné připojit maximálně 26 zařízení. FITkitů je ale možné do systému připojit více, čímž dojde k usnadnění rozvodu kabeláže ke koncovým zařízením z velkého množství elektrických vodičů na jeden síťový. Přidáním dalších FITkitu dosáhneme distribuce řídících jednotek, například na každém hlídaném patře bude jedna. Při větším počtu zařízení je také výhodné rozdělit řídící jednotky (v případě téhle práce FITkit), na jednotky obsluhující vstupní zařízení a jednotky obsluhující výstupní zařízení, čímž se dá snížit čas odezvy.

Distribucí zátěže na více FITkitů je možné odstranit přetěžování ethernetového modulu, který při vetší komunikaci začne celý systém zpomalovat.

Dalším prvkem, který je možno rozšířit, je prvek vizuální kontroly, kde může dojít maximálně k vyčerpání IP adres místní sítě. Tento problém by mohl nastat například při sledování každé místnosti ve větším městě, ale takové situaci by se s největší pravděpodobností nedokázal přizpůsobit řídící server.

Systém by se dal dále rozšířit použitím speciálních komponent běžně používaných v inteligentních domech. Mezi tyto komponenty paří speciální vypínače, které lze ovládat fyzicky i vzdáleně, dále speciální dálkově ovladatelné regulátory tepla, vody a například plynu. Pomocí těchto regulátorů, by bylo možné v případě spuštění požárního poplachu vypnout centrální přívod plynu, nebo vytvořit jednoduchý hasící systém a tím zabránit poškození větší části objektu. Pro pohodlí uživatele systému je možné vzdáleně ovládané vodovodní ventily použít k napuštění vany. Jako další zařízení by v případě vany mohly být použity vodotěsný digitální teploměr a elektronická kontrola výšky hladiny, což by spolu s vhodně vytvořeným řídícím algoritmem dokázalo napustit vodu předem požadovaného množství a teploty.

Jako další rozšíření bezpečnosti systému jej lze dovybavit GSM modulem, který může v případě narušení bezpečnosti odeslat krátkou textovou zprávu na předem nastavená čísla, například majiteli a bezpečnostní službě, nebo policii.

Takovýchto rozšíření systému by šlo použít mnoho a díky univerzálnosti dostupných zařízení a hlavně díky využití ethernetové sítě v systému, je možné v dnešní době k systému připojit velké množství běžně používaných zařízení.

8 Závěr

Cílem této bakalářské práce bylo vyrobit a naprogramovat funkční model vzdálené správy domácnosti. Tato problematika se v dnešní době začíná hodně rozšiřovat, ale cena je stále dost vysoká. Tato práce nabízí jednoduchou alternativu k inteligentním domům nabízeným na trhu za nesrovnatelně nižší cenu.

Seznámil jsem se s dostupnými technologiemi, které se běžně využívají při vzdálené správě a vzdáleném ovládání budov a z těchto technologií jsem vybral ty, které se nejvíce hodily pro prezentaci vytvořené práce na modelu.

Byl vytvořen fyzický model navrženého systému. Tento fyzický model sloužil k otestování funkčnosti systému. V budoucnosti může sloužit k prezentaci funkčnosti systému. Model byl vytvářen pro účely testování vytvořeného systému, takže odpovídá původně navržené mapě objektu. V modelu je použita většina zařízení z návrhu. Nebyl použit pouze mikrospínač a elektromagnetické relé. Mikrospínač nebyl použit z důvodu obtížné montáže do minimalizované verze modelovaného objektu. Elektromagnetické relé, které spíná obvod s nebezpečným dotykovým napětím, nebylo využito kvůli bezpečnému používání modelu.

Systém je navržen pro budoucí rozšíření o více řídících jednotek, což by teoreticky mohlo odstranit problém s přehříváním ethernetového modulu, když je připojena pouze jedna plně vytížená řídící jednotka. Rozšiřitelnost souvisí hlavně s počtem připojených zařízení, který je v podstatě neomezený.

Literatura

[1] WANG, Shengwei. *Intelligent buildings and building automation*. New York: Spon Press, 2010, 248 s. ISBN 978-0415475709.

[2] Principy detekce vniku nežádoucích osob do objektu. In: *Radomír Kaisler - SLABOPROUDY.CZ* [online]. 2002 [cit. 2012-04-01]. Dostupné na URL: <http://www.slaboproudy.cz/index.php/seznclа/82-pdvnoo020611>.

[3] Ceiling Detector. In: *Longhorn* [online]. [2010] [cit. 2012-04-07]. Dostupné na URL: <http://www.securityalarmcn.com/1-2-3-2-2-ceiling-detector.html>.

[4] LONGHORN. *Manuál F-PIR-ČIDLO PDS-901A* [online]. [2007] [cit. 2012-05-07] Dostupné na URL: <http://www.gme.cz/dokumentace/754/754-183/orn.754-183.1.pdf>.

[5] N-SA03B. In: *GM LECTRONIC* [online]. [2007] [cit. 2012-05-07]. Dostupné na URL: <http://www.gme.cz/magneticke-kontakty/n-sa03b-p630-166/>.

[6] Mikrospínač ZIPPY DM-03S-2P-Z. In: *GM ELECTRONICS* [online]. [2007] [cit. 2012-05-07]. Dostupné na URL: <http://www.gme.cz/mikrospinace-do-dps/mikrospinac-zippy-dm-03s-2p-z-p630-033/>.

[7] Síťová IP kamera Ubiquiti airCam 1 Mpix. In: [online]. [2010] [cit. 2012-04-07]. Dostupné na URL: <http://www.mujinternet.eu/index.php?page=shop.product_details&flypage=flypage-vmblend.tpl&product_id=82&category_id=12&vmcchk=1&option=com_virtuemart&Iteмte=1>.

[8] VAŠÍČEK, Zdeněk. *Úvod - FITkit* [online]. 2006 [cit. 2012-02-01]. Dostupné na URL: <http://merlin.fit.vutbr.cz/FITkit/>.

[9] GILMORE, W et al. *Velká kniha PHP 5 a MySQL*: *kompendium znalostí pro začátečníky i profesionály*. Nové, 3. vyd. Překlad Jan Pokorný. Brno: Zoner Press, 2011, 736 s. Encyklopedie Zoner Press. ISBN 978-80-7413-163-9.

[10] PROCHÁZKA, David. *CSS a XHTML*: *tvorba dokonalých WWW stránek krok za krokem*. 2., aktualiz. vyd. Praha: Grada, 2011, 175 s. Průvodce (Grada). ISBN 978-80-247-3897-0.

[11] GRUDL, David. *Nette Framework* [online]. 2008 [cit. 2012-04-15]. Dostupné na URL: <http://nette.org/cs/>.

[12] HEROUT, Pavel. *Učebnice jazyka C*. 6. vyd. České Budějovice: Kopp, 2009. ISBN 978-80-7232-383-8.

[13] FRED, Halsall. *Computer networking and the internet*. Vyd. 1. Edinburg: Addison-Wesley, 2005, 803 s. ISBN 03-212-6358-8.

[14] ODELL, Den. *JavaScript*: průvodce *programováním ajaxových aplikací*. Vyd. 1. Brno: Computer Press, 2010, 368 s. ISBN 978-80-251-2733-9.

[15] Automatická detekce vzniku požáru 1. díl. In: *Radomír Kaisler - SLABOPROUDY.CZ* [online]. 2002 [cit. 2012-04-01]. Dostupné na URL: <http://www.slaboproudy.cz/index.php/seznc la/80-advp1021119>.

[16] Detektory PIR 1. díl. In: *Radomír Kaisler - SLABOPROUDY.CZ* [online]. 2002 [cit. 2012-04-01]. Dostupné na URL: <http://www.slaboproudy.cz/index.php/seznc la/86-pujsi1>.

[17] Netcam.cz. *Co je to IP* kamera? [online]. [2009] [cit. 2012-05-09]. Dostupné na URL: <http://www.netcam.cz/encyklopedie-ip-zabezpeceni/co-je-sitova-kamera.php>.

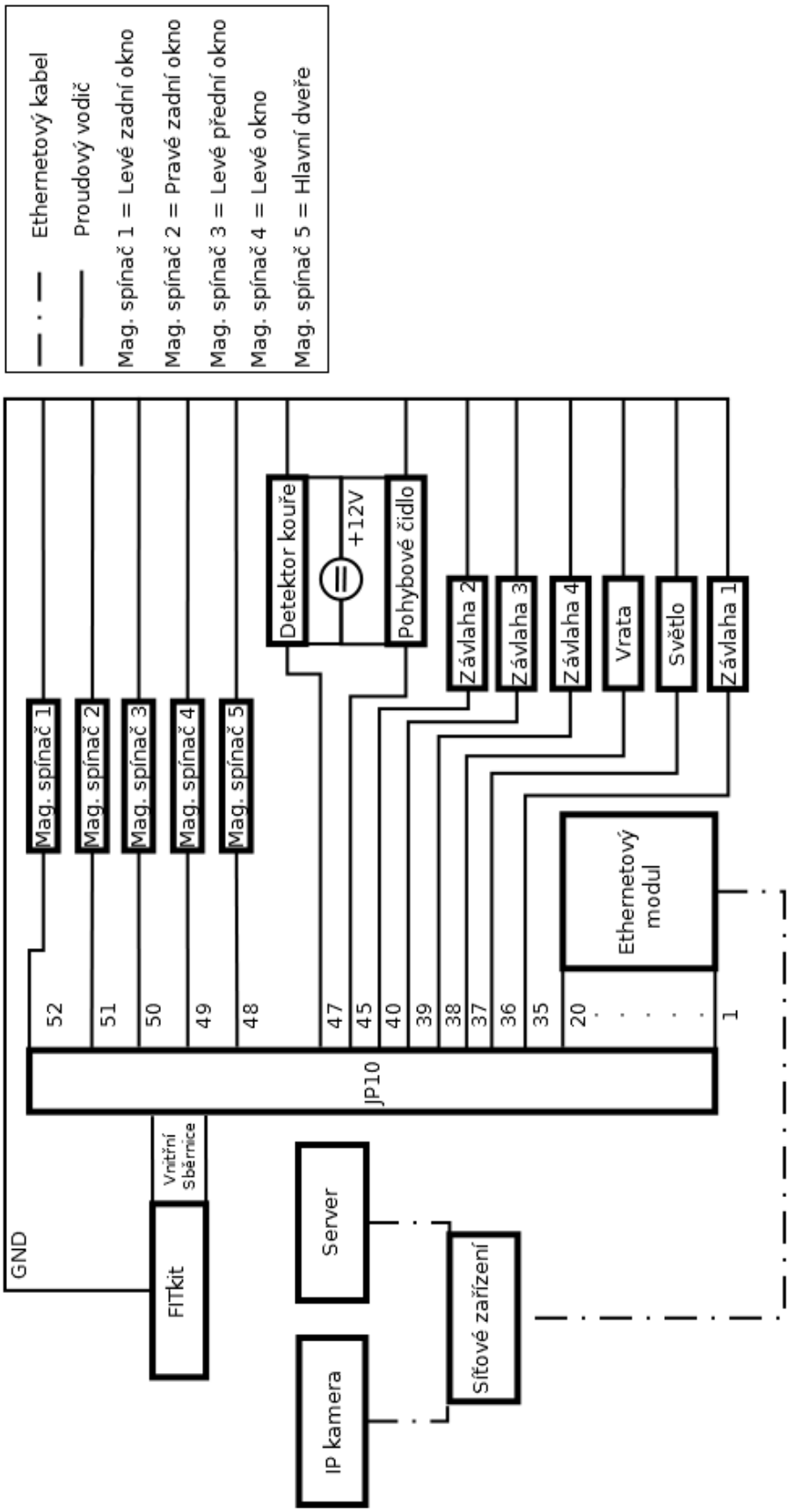
[18] *RFC 2818 - HTTP Over TLS* [online]. 2000 [cit. 9.5.2012].]. Dostupné na URL: <http://tools.ietf.org/html/rfc2818>.

[19] MAŤÁTKO, Jan. *Elektronika*. 5. vyd., V [nakl.] Idea servis 3., rozš. vyd. Praha: Idea servis, 2002, 327 s. ISBN 80-859-7042-2.

Seznam příloh

Příloha 1. Schéma zapojení vyrobeného modelu
Příloha 2. Návod k sestavení a použití modelu
Příloha 3. Fotky modelu
Příloha 4. DVD s požadovanými soubory, jejichž struktura je detailně popsána v souboru README.txt

Příloha 1: Schéma zapojení modelu



Příloha 2:
Návod k instalaci aplikace

1. Na počítači zvoleném jako server celého systému je nutné zkopírovat soubory webové aplikace do vlastní složky. Dále je nutné zkopírovat zdrojové kódy pro FITkit do složky, ve které je očekává software programující FITkit (v případě použití QDevKit se tato složka jmenuje svn).
2. Nyní musíme nainstalovat webový server a vytvořit instanci pro aplikaci umístěnou ve vybrané složce (aplikace byla vyvíjena a testována na webovém serveru IIS verze 6.1), vygenerovat si vlastní certifikát a nastavit použití https protokolu. Dalším krokem je nastavení instance jako výchozí. Toto nastavení je možné ověřit zadáním slova *localhost* do příkazového řádku webového prohlížeče.
3. Ve složce src je uložen soubor create_database.sql, který musíme spustit nad databází. Po spuštění tohoto skriptu je nutné upravit v souborech s webovou aplikací soubor app/config.ini vazby na databázi.
4. Následně je potřeba povolit obousměrnou komunikaci https přes firewall, nebo firewall úplně vypnout.
5. Nyní vytvoříme ethernetovou síť pomocí síťového zařízení. K sítí připojíme pomocí ethernetových kabelů FITkit, server a IP kameru, kterou jsme před tím zapojili podle návodu dodaného výrobcem
6. Nyní nastal čas sestavení fyzických částí podle přiloženého schématu. Schéma popisuje připojení ethernetového modulu k JP10 sběrnici FITkitu a také připojení modulu se zařízeními, který lze vytvořit na nepájivém poli a připojit pomocí kabelů osazených koncovkami, také k JP10 sběrnici. Zařízení zapojujeme jedním kontaktem k minusovému napětí na FITkitu a druhým k portu popsaném ve schématu. Zařízení, která potřebují externí napájení (senzor kouře a pohybu), zapojíme dle pokynů výrobce.
7. V tomto kroku připojíme FITkit pomocí USB kabelu k serveru, spustíme program pro naprogramování FITkitu (QDevKit), naprogramujeme FITkit a spustíme ho. Poté co FITkit naběhne, vypíše nastavenou IP adresu, kterou musíme přepsat do souboru default.map.latte umístěné v souborech s webovou aplikací umístěné app/templates.
8. Posledním krokem je spuštění webového prohlížeče (nejlépe google chrom), spuštění stránky localhost a zadáním uživatelského jména a hesla (login: admin, password: admin)

Návod k použití aplikace

1. Po zapnutí aplikace a přihlášení máme na levé straně formulář pro zadávání časového nastavení a spouštění podle teplotního rozsahu. U časového spouštění je možné zadat časy ve tvaru hh:mm:ss, nebo nechat pole prázdné pro vypnutí časovače. U spouštění podle teploty je nutné zadat celé číslo v rozsahu od -99 do +99, nebo prázdné pole pro vypnutí.
2. V prostřední části je mapa objektu, kde jsou fyzicky ovládaná zařízení překreslována podle aktuálního stavu a vzdáleně ovládaná zařízení, která se ovládají stiskem příslušného obrázku či bloku.
3. V pravé části je seznam změn stavů zařízení, v období dané daty nad výpisem. Výběr data se provádí pomocí vyskakovací tabulky nebo musí být zadány ve tvaru YY:mm:dd.

Příloha 3: Fotky modelu

Foto: Boční pohledy.

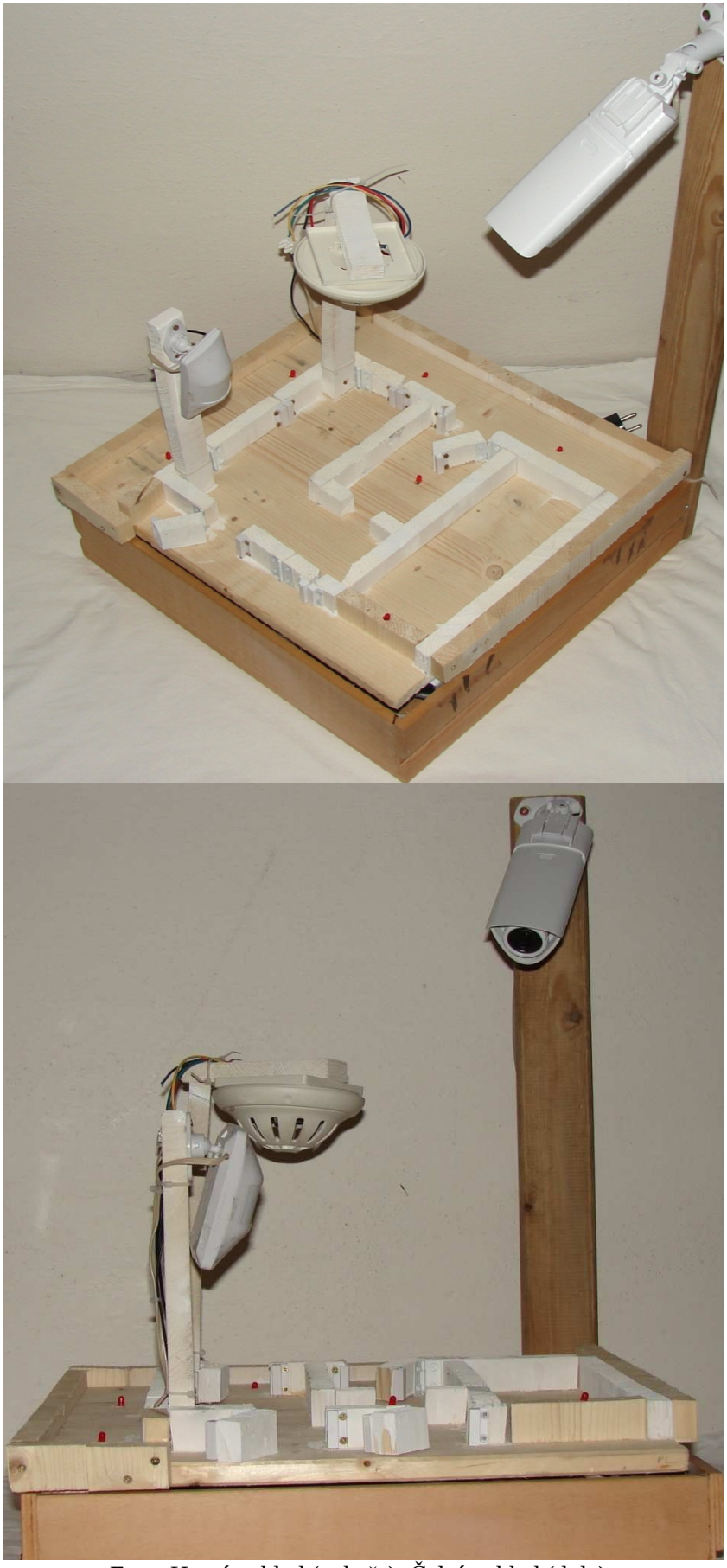


Foto: Horní pohled (nahoře). Čelní pohled (dole).